Introduction

This document describes the Motor Control Kit (YRMCKITRL78G14) based on RL78/G14.

This platform drives a 3-phase Permanent Magnet Synchronous Motor (Brushless Motor) by using an advanced sensor-less Field Oriented Control algorithm.

The phase currents measurement is done via three shunts which offer a very low cost solution avoiding any expensive current sensor.

The main focus applications for this type of algorithm are compressors, air conditioning, fans, and so on.

The platform allows easy custom applications development, including an on-board small motor and a user-friendly PC graphical user interface.

The platform can be powered directly by the PC USB interface or via an external power supply and in this case the USB communication can be optically isolated.

An external power stage can be managed, in order to control higher power motors.

Target Device

RL78/G14 series

Contents

1. Installation and Set-Up 4

1.1. Software Installation 4
1.1.1 Virtual UART USB Driver Installation on Windows XP 5
1.1.2 Virtual UART USB Driver Installation on Windows 7 8
1.1.3 Confirmation of USB Driver Installation 10

1.2. Stand Alone Demonstration Setup 11

1.2.1. Motor Connections 11

1.2.2. Set the Power Supply Selection 11

1.2.3. Set the USB Interface Connection 11

1.2.4. Select the Internal Inverter 12

1.3. Operating the Demonstration 12

1.3.1. Connecting the board 12

1.3.2. Stand Alone Demonstration Mode Operation 13

1.3.3. Demonstration Mode using the GUI 13

2. Hardware Description 14

2.1. Power Supply Selection 16

2.2. LED Description 17

2.3. Communications / Debug / Programming Interface Jumper Management 18

2.4. Internal/External Power Stage Selection 20

2.4.1. Internal Power Stage Description 21

2.4.2. External Power Stage Interface 22

2.4.2.1. External Power Stage Interface Signal Definitions 23

3. Control MCU overview 24

4. RL78/G14 Motor Control Kit Specifications and Performance Data 25

5. PC User Interface (GUI) 26

5.1. Launching the PC User Interface (GUI) 26

5.2. Function Button Description 29

5.2.1. Algorithm Selection 29
5.2.2. Parameter Setting 30
5.2.3. System Monitor 30
5.2.4. Main Control Window (Speed Control) 31
5.3. GUI Control Command Flow 35
6. Motor Calibration using the PC GUI Interface 39
6.1. Stator Phase Resistance 39
6.2. Start Up current 40
6.3. Maximum Current 40
6.4. Synchronous Inductance 41
6.5. Tuning the initial Current Pi Gains 42
6.6. Tuning the Speed Pi Parameters 44
7. Permanent magnets brushless motor model 45
8. Sensorless F.O.C. algorithm 49
9. Software Description 50
10. Start-up Procedure 53
11. Reference System Transformations in Detail 55
12. PWM Modulation Technique 56
13. Internal Representation of Physical Quantities 57
14. List of variables used in “motorcontrol.c” 58
15. Application Customisation 61
16. Renesas Flash Programmer Usage 62
17. IAR Embedded Workbench Usage 70
18. Online technical support and information 78
Revision History 79
General Precautions in the Handling of MPU/MCU Products 80

1. Installation and Set-Up

The following section provides the information to install the software, projects and documentation for the YRMCKITRL78G14 motor control kit and to be able to set up the hardware in order to run the built in demonstration.

1.1. Software Installation

To install the software supplied with the kit, Place the CD-ROM into the PC drive

The installation should start automatically
If for any reason the CD fails to start automatically, please run the "setup.exe" file on the CD

Windows™ Vista and 7 users may see "User Account Control" dialog box. If applicable, enter the administrator password and click <OK>. It is recommended that the user has sufficient administration rights to install the software on the CD. The following shall be installed:

The following software shall be installed or copied onto the user's PC

- Motor Control GUI
- IAR Embedded Workbench including documentation
- Renesas Flash Programmer including documentation
- RL78 IEC Self-Test Software (Source code) including documentation
- YRMCKITRL78G14 IAR Embedded Workshop motor control project
- Renesas Manual Navigator
- Virtual UART USB Drivers

Please note that the IAR Embedded Workbench will require license registration. Please follow the instructions during installation and ensure no other instance of the IAR Embedded Workbench is open.

The YRMCKITRL78G14 documentation shall be copied during installation and is designed to be viewed with the Manual Navigator which is installed with this kit.

The following documentation shall be copied onto the user's PC

- RL78/G14 user manual
- uPD0730 USB user manual
- RJK0654DPB Power MosFet user manual
- YRMCKITRL78G14 motor control kit user manual (This document)
- YRMCKITRL78G14 design files and schematics
- YRMCKITRL78G14 Quick start guide
- 15V BLDC Demonstration motor data sheet
- E1 On chip debug/programmer user manual
- Environmental documents (WHEE, RoHS)

Before any of the software can be used the Virtual UART USB drivers need to be installed
Two copies of the drivers are included in the installation process

- YRMCKITRL78G14 kit installation
- During the IAR Embedded Workbench Installation

Depending on the Windows operating system, the drivers should be used as follows

For 32bit Windows OS, 32bit USB drivers should be used (i.e. Windows XP, Vista and W7) and are located here

C:\Program Files\Renesas\YRMCKITRL78G14\Virtual UART\32bit

if the YRMCKITRL78G14 installation is used

C:\Program Files\IAR Systems\Embedded Workbench 6.0\rl78\drivers\renesas\VirtualCOM\32bit

if the IAR installation is used

For 64bit Windows OS, 64bit USB drivers (i.e. Windows Vista and W7) and are located here

C:\Program Files\Renesas\YRMCKITRL78G14\Virtual UART\64bit

if the YRMCKITRL78G14 installation is used

C:\Program Files\IAR Systems\Embedded Workbench 6.0\rl78\drivers\renesas\VirtualCOM\64bit

if the IAR installation is used

1.1.1 Virtual UART USB Driver Installation on Windows XP

When the *YRMCKITRL78G14* board is connected to the host machine, the board is recognized by "Plug and Play", and the wizard for finding new hardware is started.

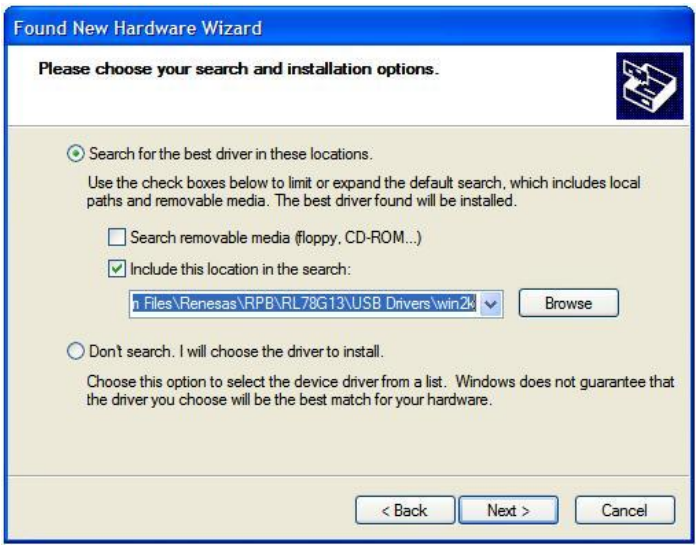
Select "No, not this time" and click "Next"

Select "Install from a list or a specific location (Advanced)" and click "Next"

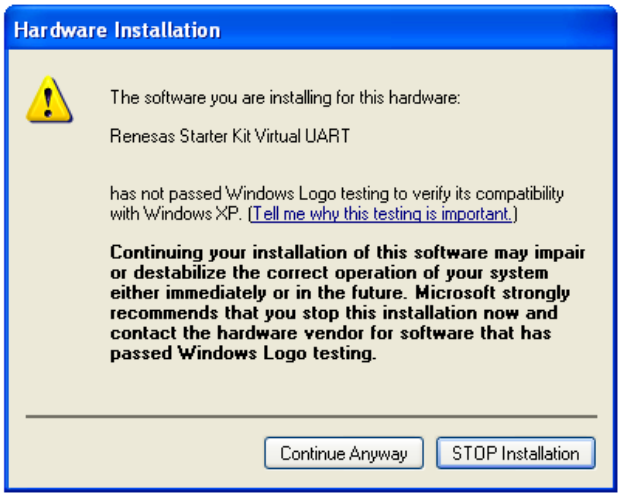
On Windows Vista, a window should also pop up with similar options when connecting the *YRMCKITRL78G14* board for the first time. Select “Locate and install driver software” and then “Browse my computer for driver software (advanced)”.

Set “Include this location in the search” and then browse the computer to select the directory indicated previously for the appropriate operating system location.

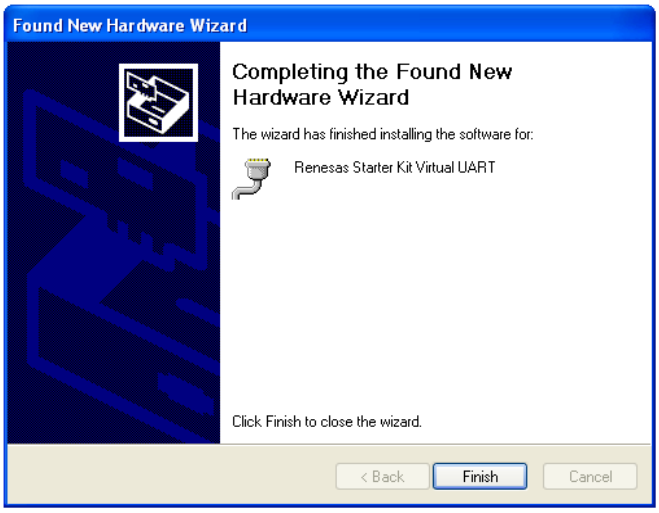
Then click “Next”



If this window appears, click “Continue Anyway” to continue the installation.



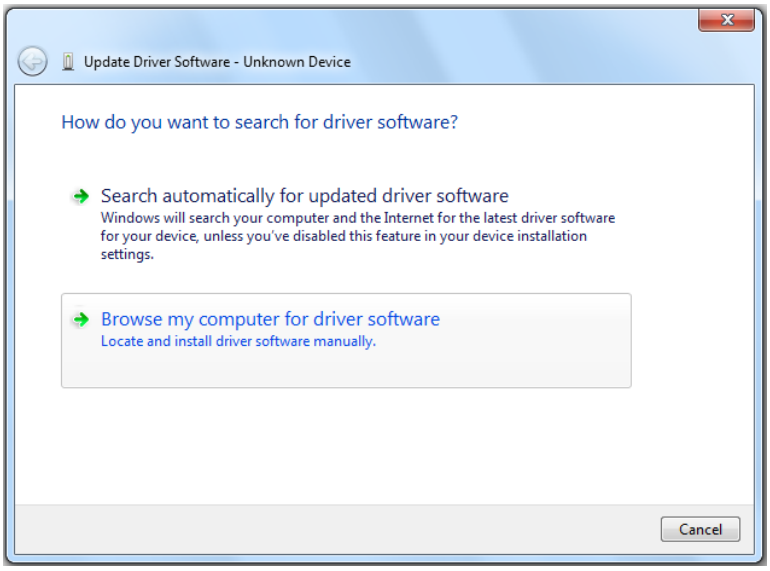
Then click “Finish” to complete the installation wizard.



1.1.2 Virtual UART USB Driver Installation on Windows 7

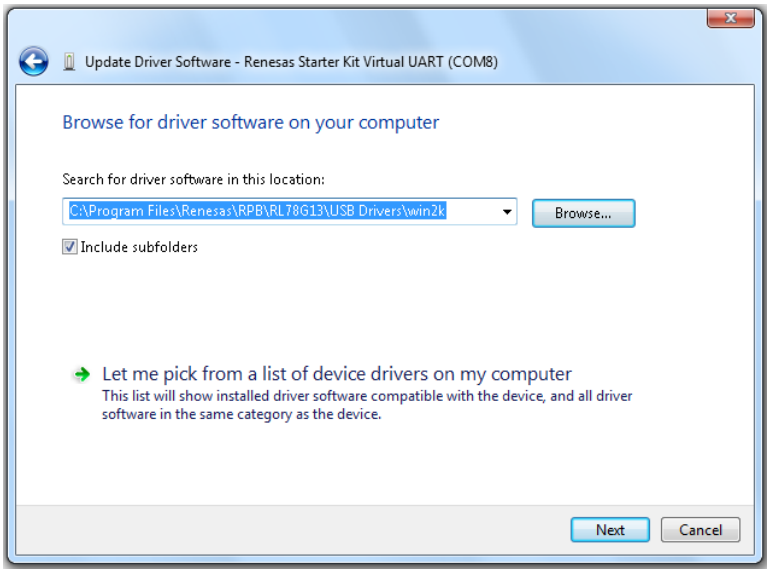
When the *YRMCKITRL78G14* board is connected to the host machine, the board is initially recognised as an "Unknown Device" in the Device Manager. Right click on the "Unknown Device" and select "Update Driver Software…" within the Device Manager window.

Select "Browse my computer for driver software".

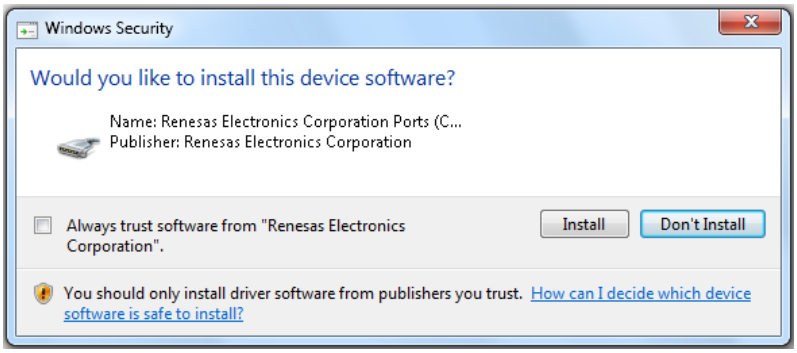


Please select the correct file location (32bit / 64bit) as indicated previously for the appropriate Operating system

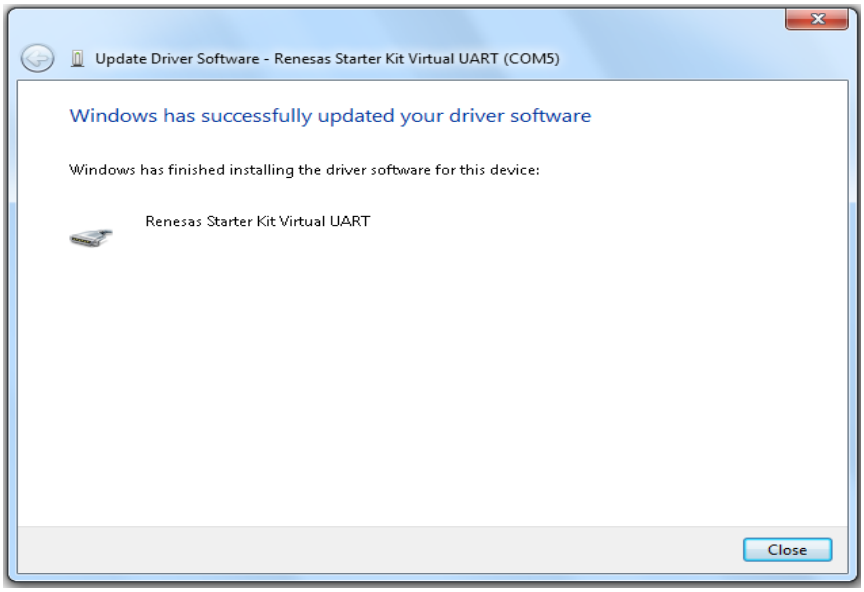
Then click "Next"



Click “Install” when the driver has been found..



Click “Close” when the installation is complete



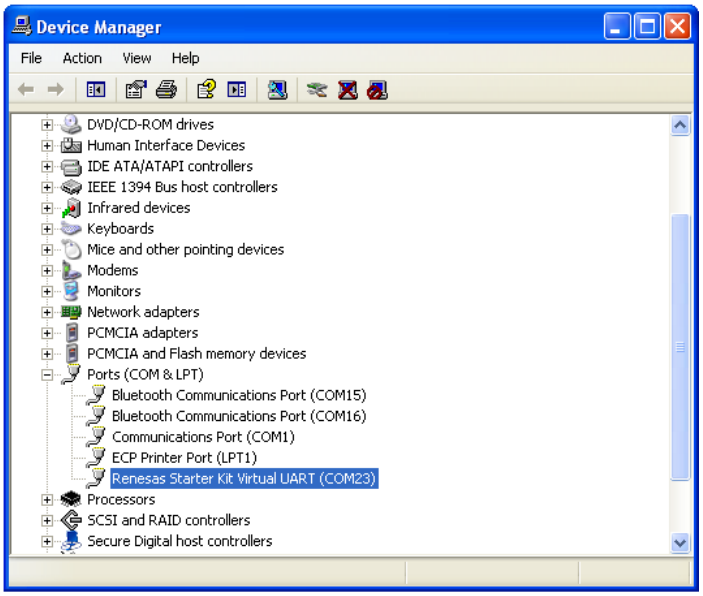
1.1.3 Confirmation of USB Driver Installation

After installing the USB driver, check that the driver has been installed correctly, according to the procedure below.

Open the "Device Manager" (this will vary depending on the Windows Operating system used)

When the *YRMCKITRL78G14* board is connected to the host PC, the “Ports (COM & LPT)” section should show the "Renesas Electronics Starter Kit Virtual UART”.

The screen below shows that the COM port number is "COM23". Note that if the board is connected to a different USB port connection, the COM port number will change.



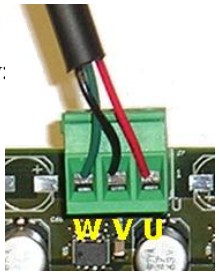
1.2. Stand Alone Demonstration Setup

The following information provides the initial setup for the YRMCKITRL78G14 motor control kit basic “out of the” box” demonstration which can be used to check that the system is working correctly and provide a quick demonstration for customers, exhibitions etc.

The following describes the default settings which configures the board for use with the USB supply and the internal Inverter. Please ensure that no external power stage is connected to connector J10

1.2.1. Motor Connections

Connect the demonstration motor supplied in the kit, to the connector as shown below:



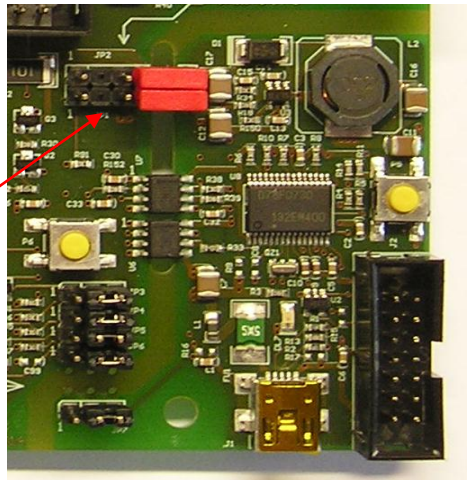
1. Red Motor wire to “U” connection
2. Black motor wire to the “V” connection
3. Green motor wire to the “W” connection
4. The Hall sensor wires should not be connected

Once connected, plug the motor connector into the board (J7)

1.2.2. Set the Power Supply Selection

Check that jumpers JP1 and JP2 are both connected between pin 4 and 6

Jumpers JP1 and JP2



1.2.3. Set the USB Interface Connection

Check that the following jumpers are connected as shown below.

JP3 pins 2 – 3

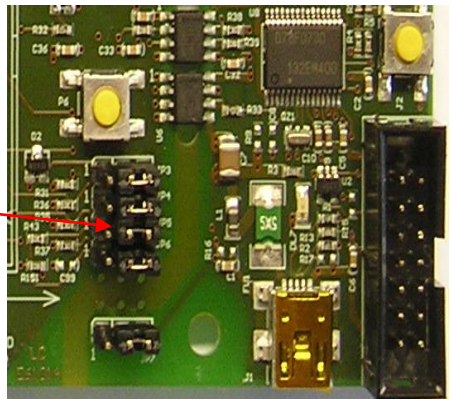
JP4 pins 2 - 3

JP5 pins 2 - 3

JP6 pins 2 - 3

JP7 pins 2 - 3

Jumpers JP3 to JP7



Note that this sets the USB interface to the “non isolated” mode

1.2.4. Select the Internal Inverter

The following jumpers should fitted to connect the power, control, current inputs and the PWM drive signals to the internal inverter

JP10 fitted

JP11 fitted

JP12 fitted

JP13 fitted

JP14 fitted

JP15 fitted

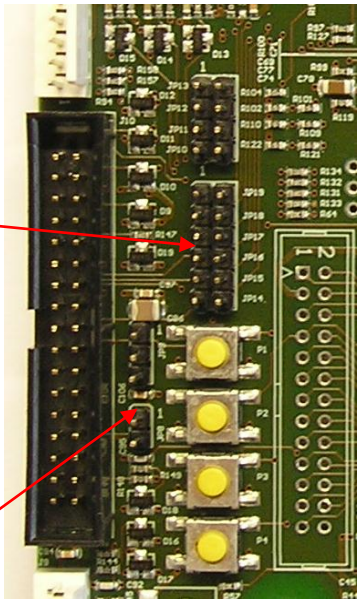
JP16 fitted

JP17 fitted

JP18 fitted

JP19 fitted

Jumper JP8 and JP9 can be connected or open



1.3. Operating the Demonstration

Once the jumpers have been checked as described above and the motor is connected
Follow the sequence described below to run the demonstration

1.3.1. Connecting the board

Connect the USB lead to the PC and then connect it to the YRMCKITRL78G14 board

The following LED should be lit (on):

DL7 USB supply from the PC and the step up regulator,

DL6 12V step-down converter output,

DL5 5V step-down converter output (logic supply).

The following LED's are controlled by the RL78G14 MCU:

DL1 should blink at approximately ½ second interval (RL78G14 running),

DL2, DL3 should be off,

DL4 should be on (it is driven with an I/O output with period equal to sampling period interrupt).

1.3.2. Stand Alone Demonstration Mode Operation

To start the demonstration sequence press the P4 button

The board should follow the following sequence:

- ⇨ The motor should start and accelerate to a steady speed
- ⇨ The motor will run at this speed and direction for approximately 10 seconds
- ⇨ The motor shall then reverse direction

Decelerate =>
Stop =>
Change direction =>
Accelerate to the previous speed =>
Continue operating for a further 10 seconds =>
The motor shall then stop

The sequence can be repeated by pressing the P4 button again.

The motor speed is defined by the Minimum and maximum speed settings.
When running from the hardware the demonstration speed is defined by the default settings in the motor control IAR project or those set during a previous session when using the GUI

The basic speed is calculated as follows:-

Demonstration Speed = Min Speed + ((Max Speed – Min Speed) / 4)

This setting of the demonstration speed equation can be changed in the code contained in the "motorcontrol.c" source file

1.3.3. Demonstration Mode using the GUI

The YRMCKITRL78G14 demonstration can also be operated as described above using the control GUI.

To start the demonstration press the "DEMO" button in the PC GUI

The demonstration will follow the same sequence as described for the stand alone operation above.

If the GUI is installed and running, the graphs will be updated with the running data (Speed, Voltage and Current) even if the demonstration is started from the push button on the board.

Note the "RPM SPEED" indicator does not operate in demonstration mode. The measured speed is updated as normal. The demonstration speed should be the same if the minimum and maximum speed settings have not been changed in the "parameter setting" window.

For further details please see the PC Control Interface section

2. Hardware Description

The YRMCKITRL78G14 starter kit is a single board motor control inverter, based on the new RL78/G14 series of microcontrollers. The hardware includes a low-voltage 3-phase MOSFET power stage, the MCU control system, the switching power supplies and the communications / debugging / programming interface.

To maximise the flexibility of the demonstration board, the following features are included:

- A complete 3-phase inverter on-board with a low voltage motor, so it becomes easy to test the sensor-less algorithm on the RL78G14.
- Two RL78/G14 devices can be supported by the board
    - The default is the 100pin device (R5F104PJAFB)
    - 64pin device (R5F104LEAFB)
    - The pin out of both RL78G14 devices are brought out to connectors JPUP1, 2, 3 and 4
- USB Communications, Debugging and Programming interface via the uPD78F0730 USB MCU.
- Connectors for hall sensors and encoder connections
- Connection for an external power supply
- On board power supply generation from either the USB or external supplies
- A user "patch area" for external signal conditioning or use of other motor control algorithms
- Sample BLDC Motor
- External power stage interface.

The default device mounted on the board is the 100pin RL78/G14 device. If required this device can be removed and a 64pin RL78/G14 device fitted

The Communication/Debugging interface allows the user to control the inverter through the graphical user interface (GUI), or to program the RL78/G14 with either the Renesas Flash Programmer or download and debug the motor control program with the development IDE debugger. This interface uses the PC USB power supply.

When using external programming or debugging tools it is possible to fully isolate the communications interface to avoid possibilities of electrical damage, especially when using the external power stage interface

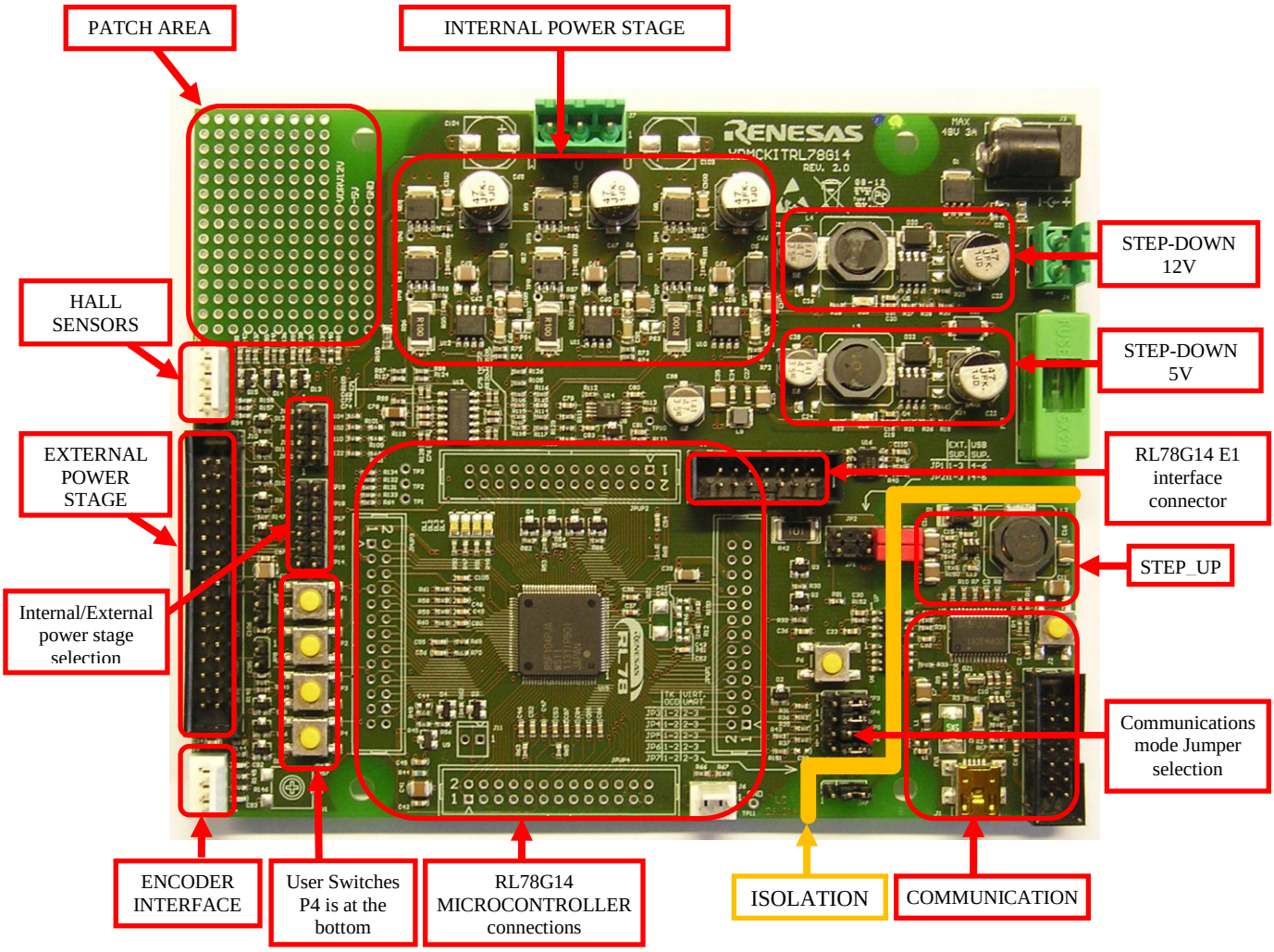
The board can be fully powered either from the PC USB interface, or from an external power supply.
When the USB power is used, a step-up converter is used to obtain the inverter VBUS voltage necessary to generate the voltages for the main board and the motor. In this mode the main system is not isolated from the PC and the current available for the motor drive is limited to approximately 300mA.

When an external power supply is used the main board is fully isolated from the PC. The power for the motor is rated at 48Vdc @3A. It may be possible to supply more motor current, but it will be necessary to provide cooling for the inverter stage power MosFets in order to keep the temperature of the power MosFets at around 60^0C

The internal power supplies (logic and for the driving system) are obtained through two step-down switching regulators, in order to reduce heating and power consumption.

A patch area was added to allow the user to make hardware modifications/adjustments as necessary. This area can also be used to support alternative motor drive algorithms, such as sensorless BLDC drive.

Please refer to the schematics for the details of the full hardware implementation.



YRMCKITRL78G14 Board Overview

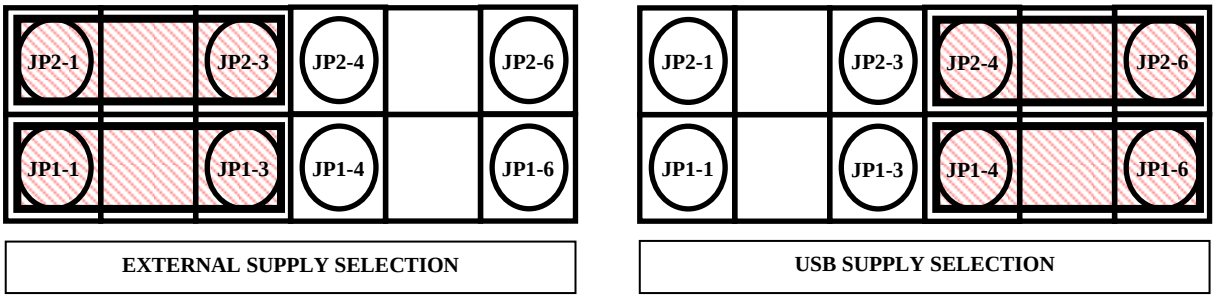
2.1. Power Supply Selection

There are two main ways to supply power to the board.

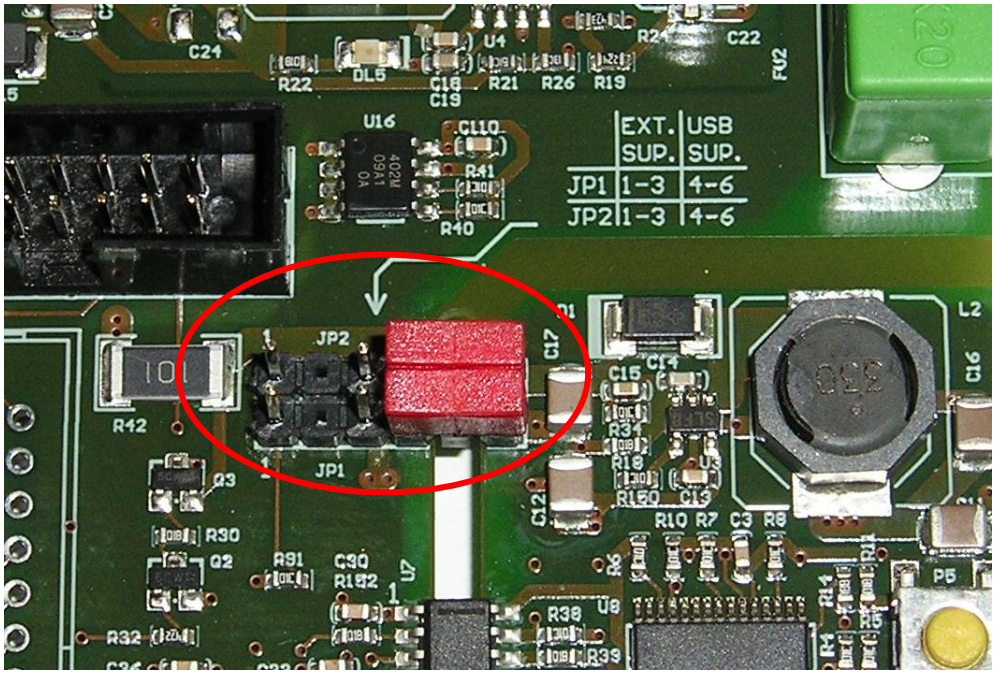
1. The first is to use the PC USB supply, and in this case the current you can give to the motor is limited by the USB interface. In this mode the USB interface is not isolated.
2. The second is to use an external voltage DC source to supply the board.
   The recommended operating voltage range is between 15Vdc and 48Vdc, max at 3Amps DC.
   A power supply with more current could be used to increase the available for the motor. However it recommended that the power MosFets are kept to a working temperature of approx. 60^0C so that some form of air cooling will be needed.

   In this setting the USB communications interface will be isolated from the inverter.

The selection between the two possibilities is made using jumpers JP1 and JP2, as shown below.



Power supply Selection Jumpers (JP1 and JP2) configuration settings



Power supply Selection Jumpers (JP1 and JP2) location

Note: The figure shown above is with the jumpers in position 3 – 4

2.2. LED Description

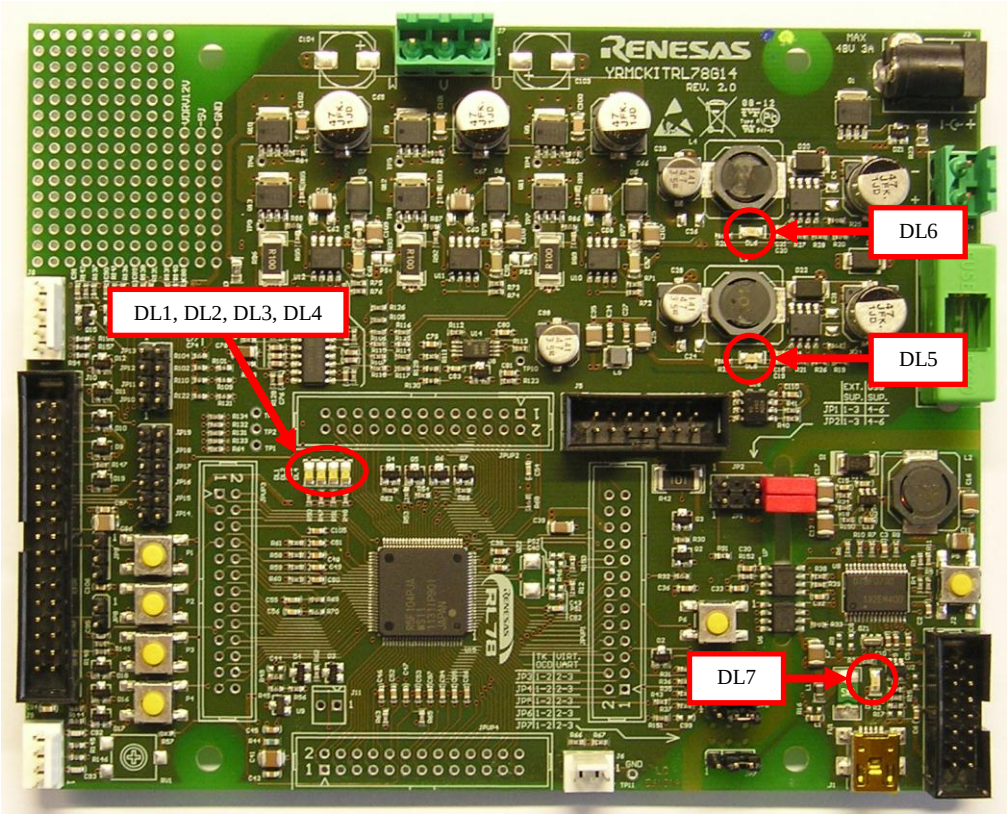
There are three LED's on the board that show the status of the power supply of the board.

- DL7 is connected to the USB supply from the PC, so it indicates that power is supplied to the USB port and, therefore the complete communication section;
- DL6 is connected to the 12V step-down converter output (VDRV12V) and indicates that the inverter drive voltage is supplied.
- DL5 is connected to the 5V step-down converter output (logic supply); it indicates that the control section is supplied.

The other LED's in the board (DL1, DL2, DL3 and DL4) are driven via software from the RL78/G14 MCU.

These have the following functions, but can be user programmable by editing the motor control project.

- DL1 - This is set to indicate that the RL78/G14 is operating
- DL2 - This is used to indicate a motor error
    - Flashes quickly when there is an error, Off otherwise
- DL3 - a Watchdog, Ram Parity, Illegal access reset is generated.
    - Is set off when a normal power on or manual reset occurs
- DL4 – Indicates the control loop interrupt timing. Lit to indicate operation
    - The LED duty will change when idle or when the motor is running. This point or I/O pin can be monitored with an oscilloscope



YRMCKITRL78G14 LED Locations

2.3. Communications / Debug / Programming Interface Jumper Management

Using the communication interface, based on the uPD78F0730 USB MCU, it is possible to run the following

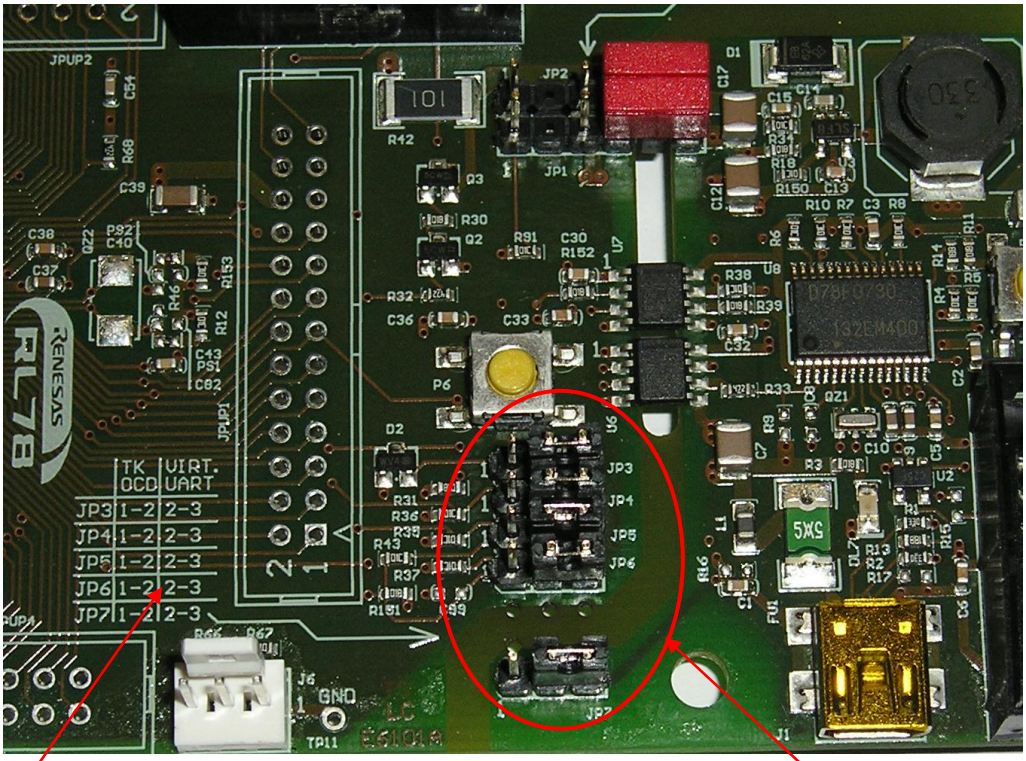
a. Control GUI,

b. Program the RL78G14 using the Renesas Flash Programmer (RFP) software

c. Debug the RL78G14 using the IAR Embedded Workbench “TK” interface.

Please note that these functions cannot be operated at the same time as they all share the same USB interface on the board. Also that it is advised that no other peripheral devices are connected to the USB port or hub when using the motor control kit.

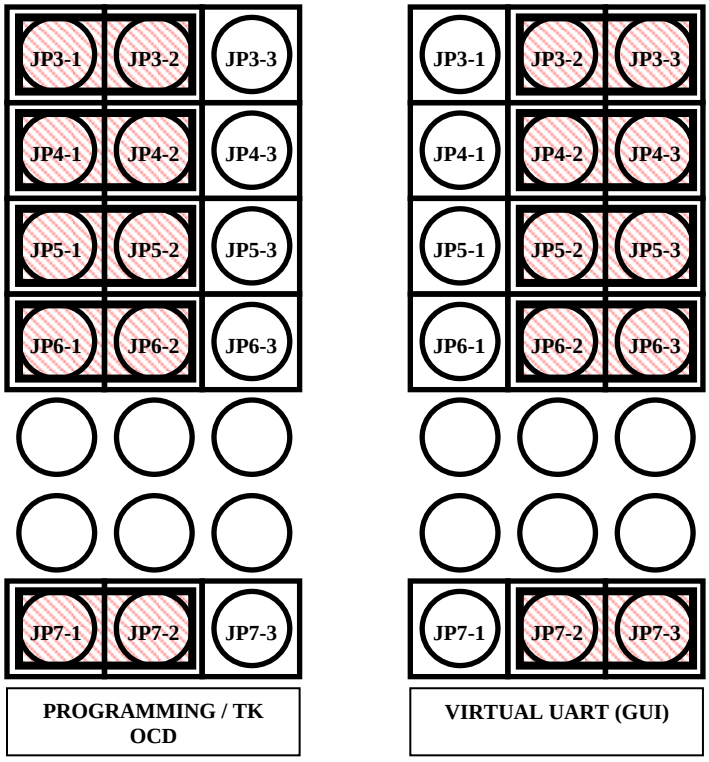
Selection of GUI or Programming/debugging is provided by jumpers (JP3, JP4, JP5, JP6 and JP7) as shown below. The settings are also included on the board for reference.



GUI / OCD Jumper (JP3, 4, 5, 6 and JP7) locations

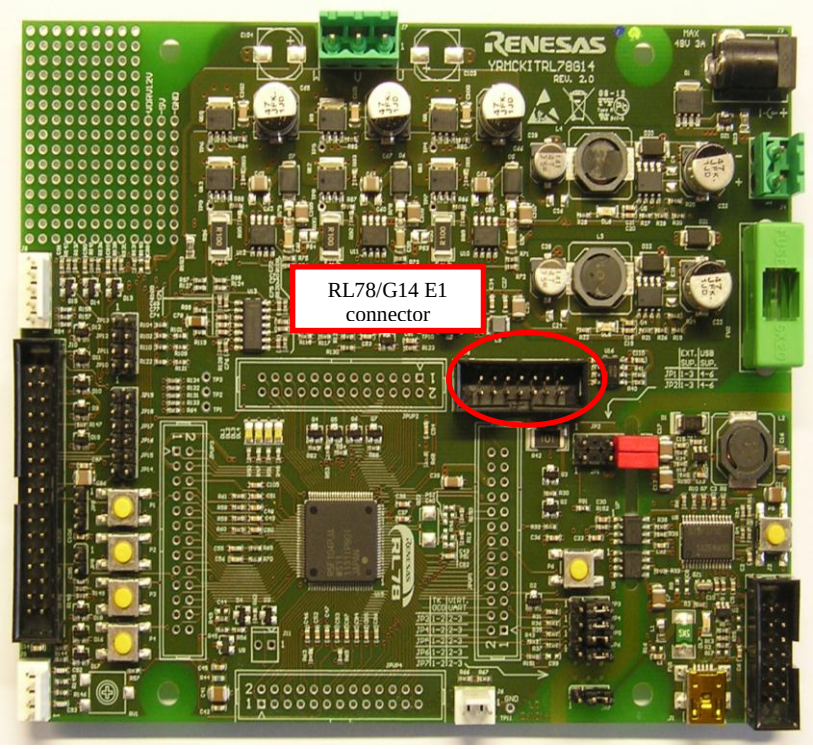
GUI or TK settings

Jumpers JP3 to JP7



GUI / OCD Jumper (JP3, 4, 5, 6 and JP7) configuration settings

NOTE: It is also possible to debug the RL78G14 using the E1 emulator, through the dedicated connector J5. Care should be made when operating from an external power supply, as this interface cannot be isolated



E1 OCD Interface connector location

2.4. Internal/External Power Stage Selection

The board offers the option of using either an internal MosFet power stage, or the complete interface to connect an external power stage.

(Please note that the internal and external interfaces cannot be operated at the same time).

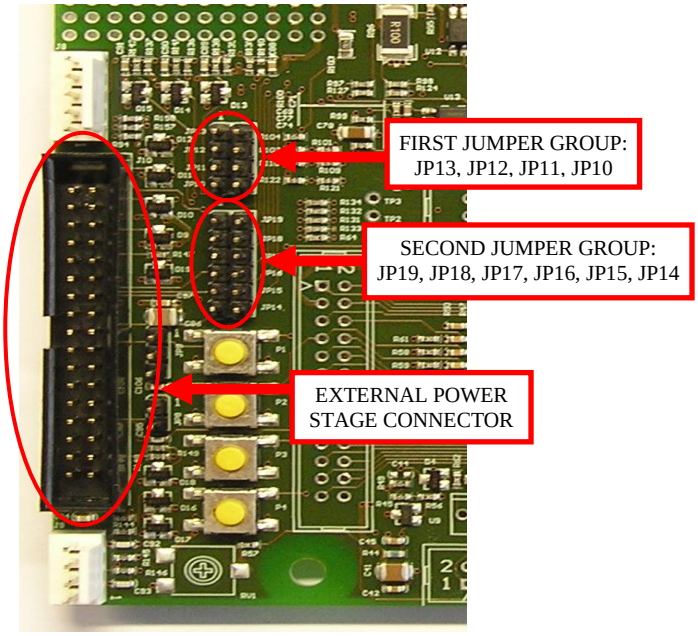
The power supplies, inverter drive signals and the other control and monitor signals are connected to the external power stage connector J10. If an external power stage is connected, then the jumpers shown below jumpers should be removed (NOT fitted).

When using the internal power stage the jumpers should be fitted and a connection to the external power stage connector must not be connected.

The jumper selections are as described below:

- First jumper group:
    - JP10 Internal U current analog channel selection
    - JP11 Internal V current analog channel selection
    - JP12 Internal W current analog channel selection
    - JP13 Internal bus voltage analog channel selection

- Second jumper group:
    - JP14 (H), JP15 (L) U phase driving signals
    - JP16 (H), JP17 (L) V phase driving signals
    - JP18 (H), JP19 (L) W phase driving signals



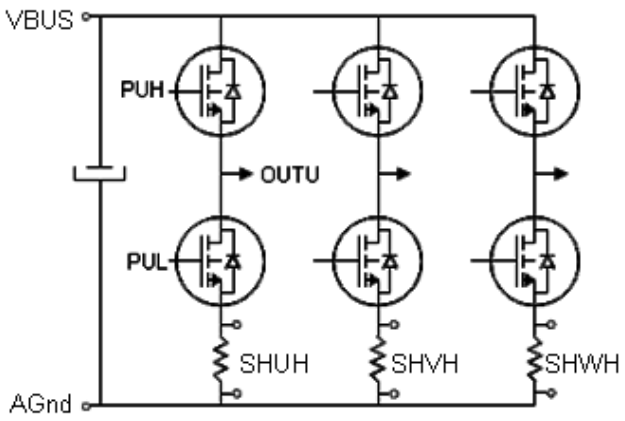
External Interface Connector and Jumper locations

2.4.1. Internal Power Stage Description

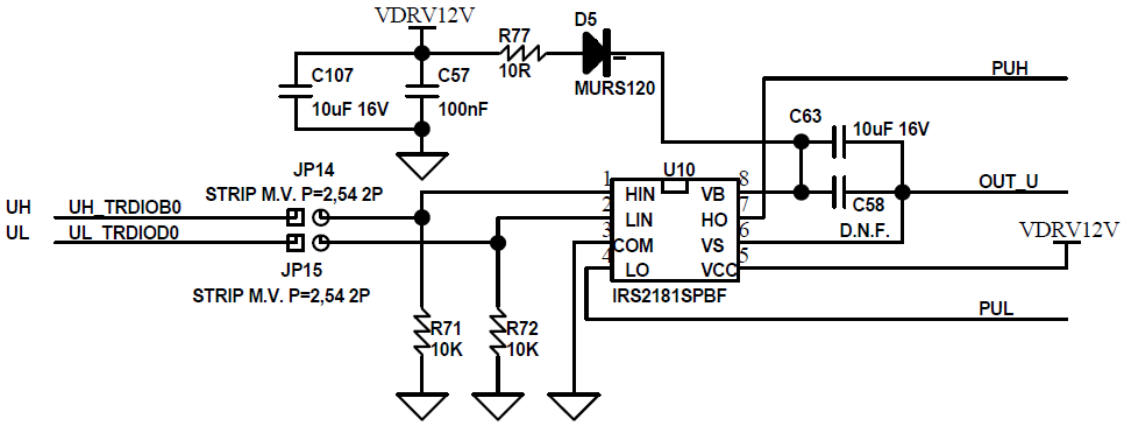
The internal power stage is forms a complete 3-phase inverter bridge using discrete low voltage power MosFets. (Renesas RJK0654DPB n-channel power MosFet)

The logic driving signals from the RL78/G14 MCU are connected to three IR MosFet drivers which take care of the level shifting and charge pump generation for the MosFet gate drive.

The inverter phase currents are read through three shunts in the low sides of the bridge.



RJK0654DPB n-channel power MosFet 3-Phase Inverter



MosFet Gate Drive Interface (One Channel)

Please refer to the board schematic for the full details.

2.4.2. External Power Stage Interface

Since internal power stage allows only the management of low voltage / low current motors, an interface with an external power stage is provided. This allows the RL78/G14 MCU together with the control GUI or IDE debugger to control a high voltage high power motor via a suitable external inverter unit.
Please find below the signal connection interface for the external power stage interface connector J10.

Temperature analog signal	1	2	Ground
Bus voltage analog signal	3	4	Ground
U current analog signal	5	6	Ground
V current analog signal	7	8	Ground
W current analog signal	9	10	Ground
Fault digital signal	11	12	U phase low switch command
Ground	13	14	V phase low switch command
Ground	15	16	W phase low switch command
Ground	17	18	U phase high switch command
Ground	19	20	V phase high switch command
Ground	21	22	W phase high switch command
Ground	23	24	SUP-1
SUP-1	25	26	SUP-2
Hall A digital signal	27	28	Hall B digital signal
Hall C digital signal	29	30	SUP-2
Encoder A digital signal	31	32	Encoder B digital signal
Encoder Z digital signal	33	34	Ground

External Interface Connector - Signal Connections

2.4.2.1. External Power Stage Interface Signal Definitions

The following provides a short explanation of the external power stage interface connections
Please refer to the circuit schematics for full details.

Analogue Signals

The analog signals are inputs and should always be in the range 0V to 5V. It may be necessary to adjust when a different motor is used and the value of the shunt resistor is changed

Drive Signals

The driving signals are outputs and are at logic (5V) level, and can be set as active high or active low
(This can be defined in the motor control project)

Alarm Signal

The alarm is an input and it is active low (no alarm level = 5V).
A pull-up is included on the board in case there is no connection to J10

Hall and Encoder Signals

The hall and the encoder signals are logic inputs and all have appropriate pull-up resistors

Supplementary Signals

Signal "SUP1", connector pins 24 and 25 can be connected to:

1. The internal power stage bus voltage (VBUS) by connecting Jumper 9 pins 1 - 2
2. The internal power stage 12V drive voltage (VDRV12V) by connecting Jumper 9 pins 2 - 3
3. Unconnected. Jumper 9 left open.

Signal "SUP-2" connector J10 pins 26 and 30 can be connected to

1. The logic supply +5V by connecting jumper JP8
2. Left unconnected

NOTE:

Care should be taken with the connections of Jumpers 8 and 9, when using the external interface as these supply the local board 5V and either the VBUS or VDRV voltages to the external power stage. It is recommended that the jumpers are removed to avoid shorting the power supplies together when an external power unit is used, unless it is necessary to either supply or these voltages to the

3. Control MCU overview

The RL78/G14 is a family of MCU's featuring the high-performance RL CPU core.

Single cycle instruction execution, with enhanced hardware support for multiply, divide and MAC operations

Below is a summary of the RL78G14 features:

RL78G14 CPU

- High-speed: 32MHz clock
- High performance: 41MIPS @32MHz
- Low current consumption: only 4.6mA @ 32MHz
- MUL, DIV, MAC hardware instructions
- Barrel shifter

MEMORY AND PACKAGES

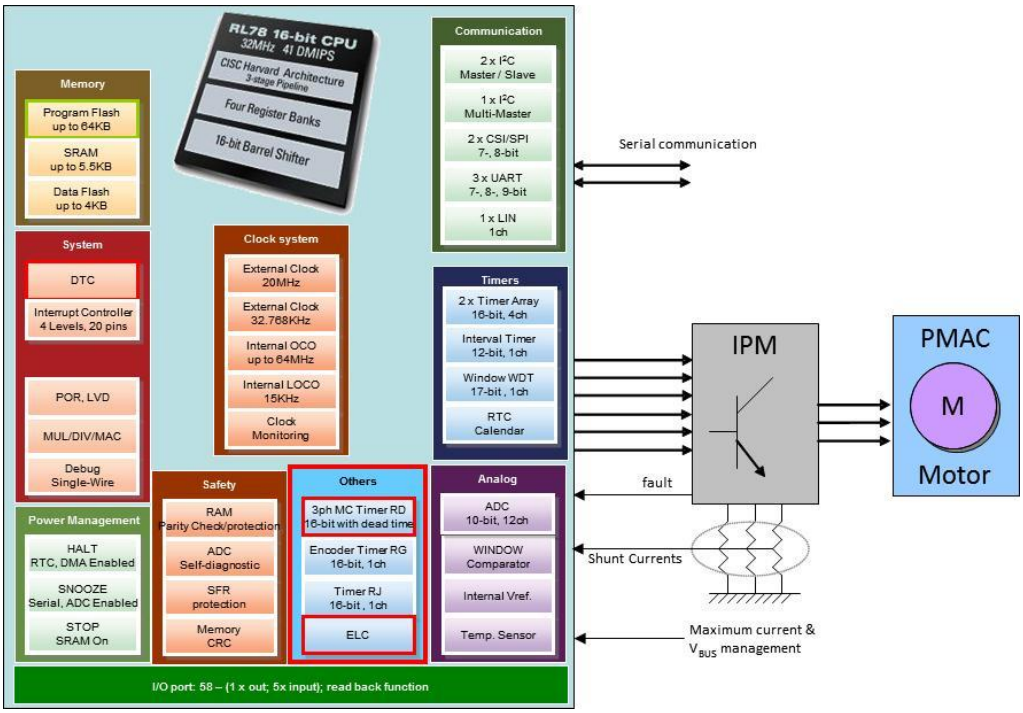
- 16KB Flash/2.5KB RAM to 256KB Flash/24KB RAM
- Up to 8KB Data Flash
- 30pin to 100pin package options

FEATURES

- 1% 64MHz internal oscillator
- 64MHz 16bit Timers
  - 3-phase 16bit PWM timer with dead time (Timer RD)
  - Multifunction timer with encoder interface (Timer RJ)
  - Timer Array Unit (TAU) and multifunction Timer (RG)
- 10-bit A/D converter up to 12 channels, 2channel 8bit DAC
- Window Comparator
- Event Link and Data Transfer controllers (ELC & DTC)
- Real Time Clock
- Independent Windowed Watchdog
- Self-Test functions (RAM/SFR Protect, ADC, System Clock Monitor, RAM Parity, Port Verify)

Large-capacity flash memory units capable of high-speed operation are included as on-chip memory, significantly reducing the cost of configuring systems.

The main application fields of this MCU include: Small household Appliances, Fans, Pumps and Power Tools.



RL78/G14 Typical Block Diagram

4. RL78/G14 Motor Control Kit Specifications and Performance Data

The specifications for the on board inverter and for the sensor-less algorithm implemented are as follows:

- External Supply voltage range: 15V to 48Vdc.
- Supply current: 3A max. (No Forced cooling)
- Maximum continuous output power 100W.
- Current reading technique: three shunt.
- Motor Control Timer Clock frequency up to 64MHz
    - PWM switching frequency: up to 24KHz.
- Sampling rate: max 8KHz.
    - Used CPU bandwidth: <70% @ 8KHz sampling rate.
- RL78G14 – R5F104PJAFB
    - 256KB Flash, 24KB RAM, 8KB Data Flash
- RL78G14 – R5F104LEAFB
    - 64KB Flash, 5.5KB RAM, 4KB Data Flash
- YRMCKITRL78G14 kit utilisation
    - Flash occupation: 15KB
    - Ram occupation: 2kB

5. PC User Interface (GUI)

The User Interface is installed automatically during the CD-ROM installation.
The PC Interface uses the USB connection to communicate with the RL78G14 board so before use please make sure that the Virtual UART drivers are installed as described in section 1.1.1

5.1. Launching the PC User Interface (GUI)

Note that for Windows Vista and Windows 7 it is necessary to run the GUI as "Run as administrator"
It is possible to set the operation to permanently enable the "Run as administrator" option in the windows start menu. To enable this option perform the following sequence

a. Right click on the "motor Control Demo" icon
b. Select Properties
c. Select Advanced
d. Click the "Run as administrator" button
e. Press "OK" twice to return

To open the PC GUI click on the "Motor Control Demo" ICON in the start menu.

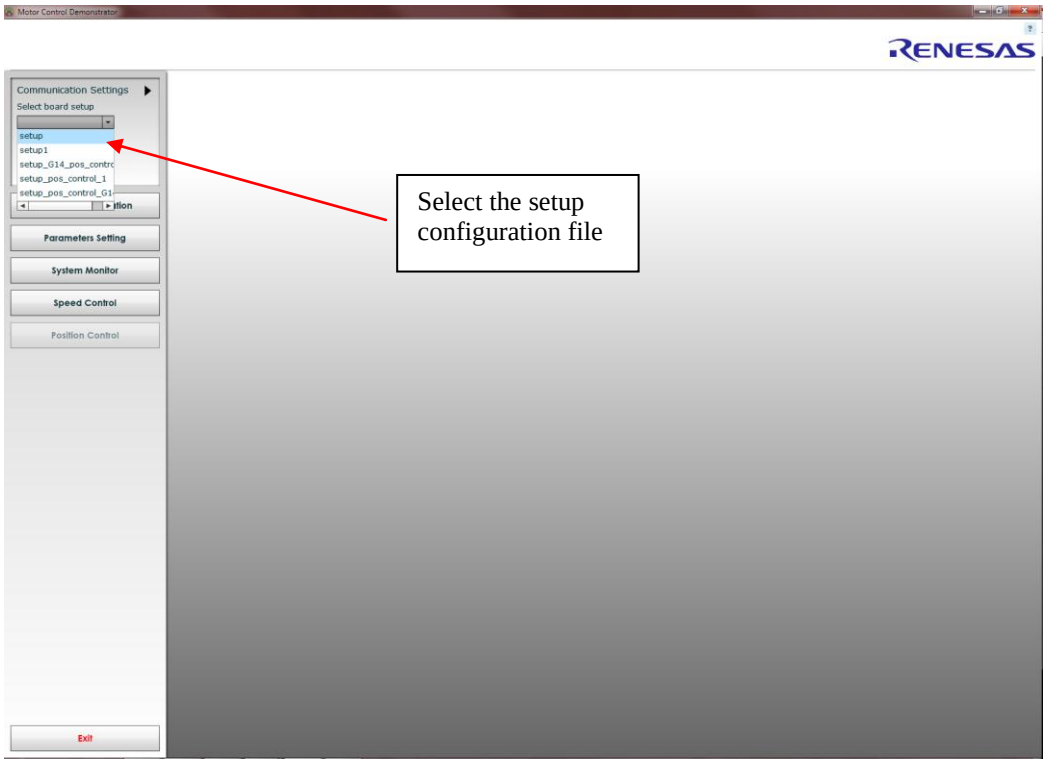
The GUI should open as shown below

To connect the GUI to the YRMCKITRL78G14 board follow the sequence

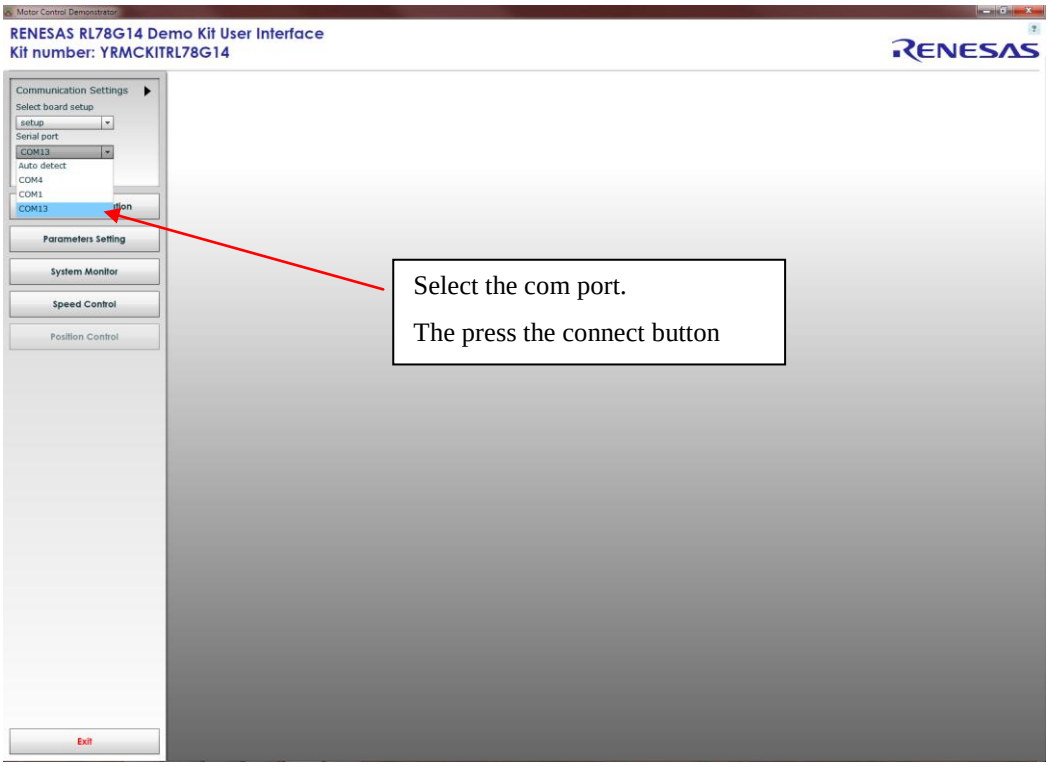
- If more than one configuration file is located in the GUI installation directory, then the user has to select the appropriate configuration (setup) file. Otherwise the default configuration file (setup.ini) is selected automatically and this part can be skipped.

  Note the setup and configuration can be customised by editing the setup.ini file and then saving as the same or alternative file name.

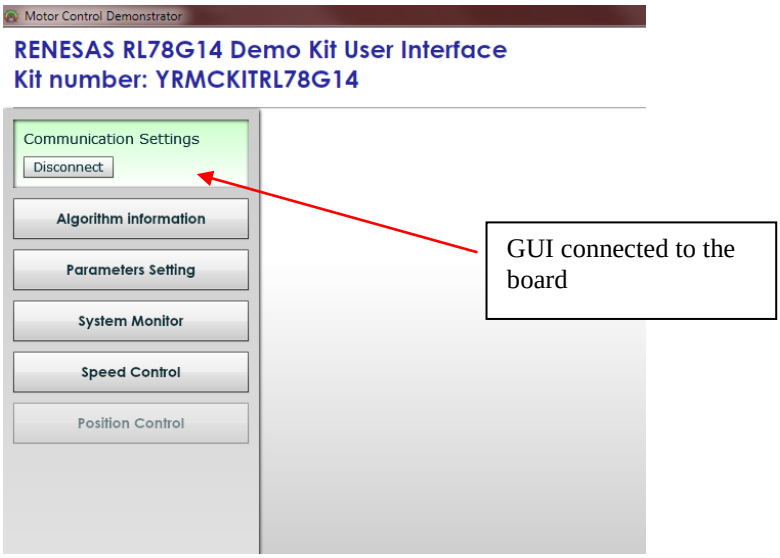
- If the setup file is not automatically selected, then first select the "Select board setup" drop down bar in the top left hand corner of the GUI panel. Then select the G14 configuration file (i.e. setup) as shown below and close the box



- Next select the “Com Port selection” drop down box and select the communications port that the YRMCKITRL78G14 board is connected to, close the drop down box and click the “connect” button as shown below



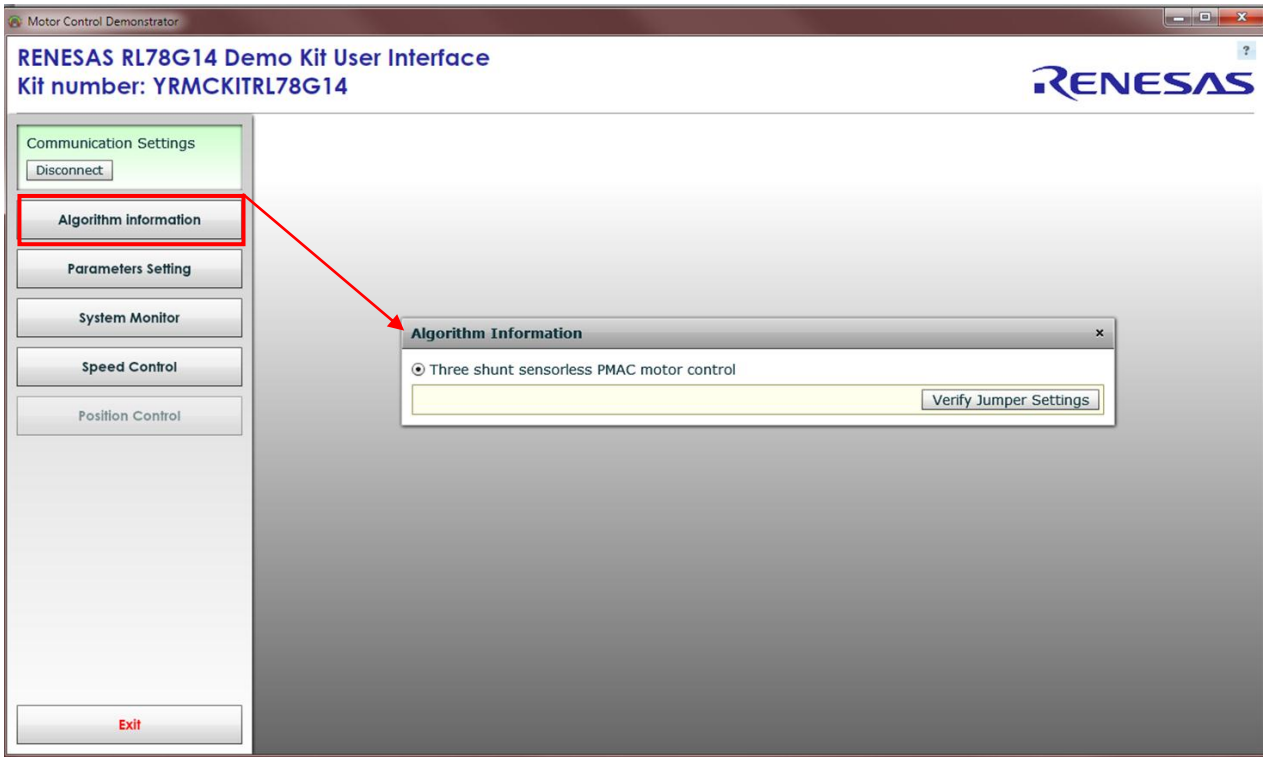
The GUI should now be connected to the board as shown below



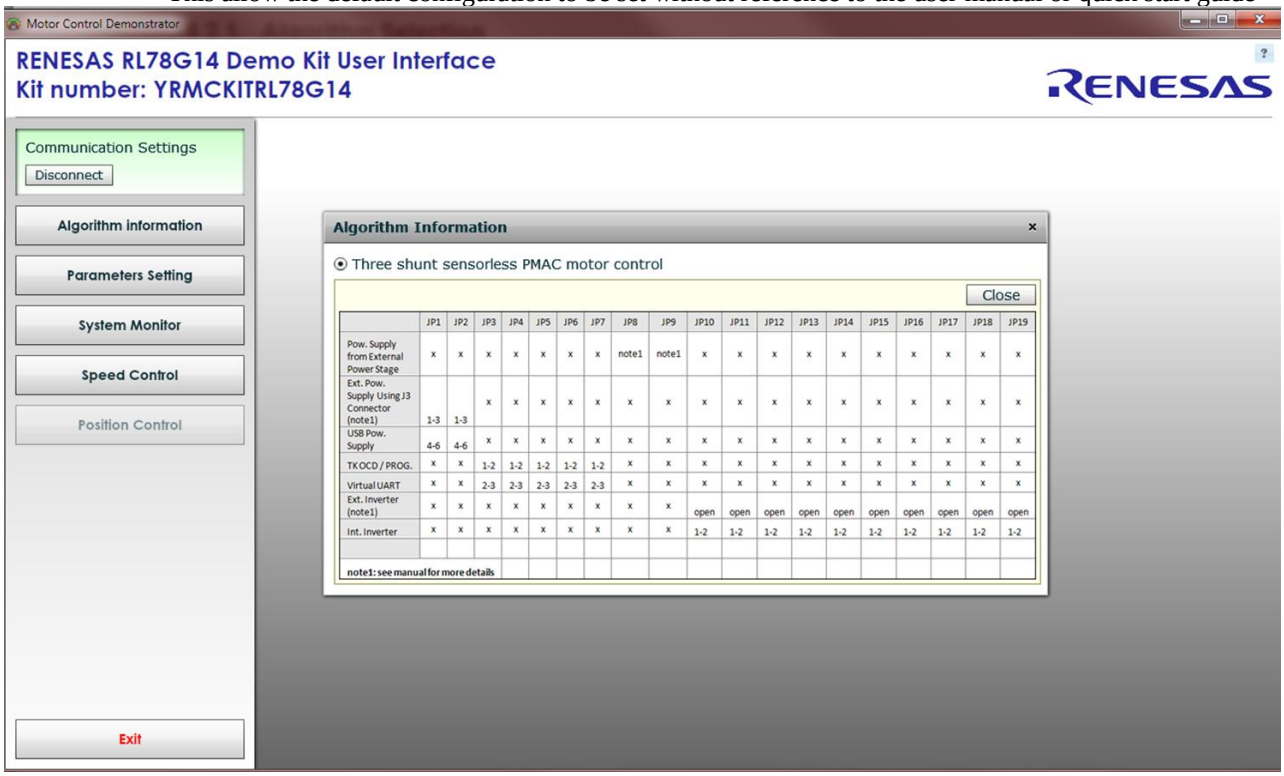
5.2. Function Button Description

5.2.1. Algorithm Selection

At this time the only algorithm available is the sensorless F.O.C algorithm Pressing the "Verify Jumper Settings" button opens the default settings for the board



Pressing the "Verify Jumper Settings" button opens the default settings for the board.
This allow the default configuration to be set without reference to the user manual or quick start guide



	JP1	JP2	JP3	JP4	JP5	JP6	JP7	JP8	JP9	JP10	JP11	JP12	JP13	JP14	JP15	JP16	JP17	JP18	JP19
Pow. Supply from External Power Stage	x	x	x	x	x	x	x	note1	note1	x	x	x	x	x	x	x	x	x	x
Ext. Pow. Supply Using J3 Connector (note1)	1-3	1-3	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
USB Pow. Supply	4-6	4-6	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
TK OCD / PROG.	x	x	1-2	1-2	1-2	1-2	1-2	x	x	x	x	x	x	x	x	x	x	x	x
Virtual UART	x	x	2-3	2-3	2-3	2-3	2-3	x	x	x	x	x	x	x	x	x	x	x	x
Ext. Inverter (note1)	x	x	x	x	x	x	x	x	x	open	open	open	open	open	open	open	open	open	open
Int. Inverter	x	x	x	x	x	x	x	x	x	1-2	1-2	1-2	1-2	1-2	1-2	1-2	1-2	1-2	1-2
note1: see manual for more details																			

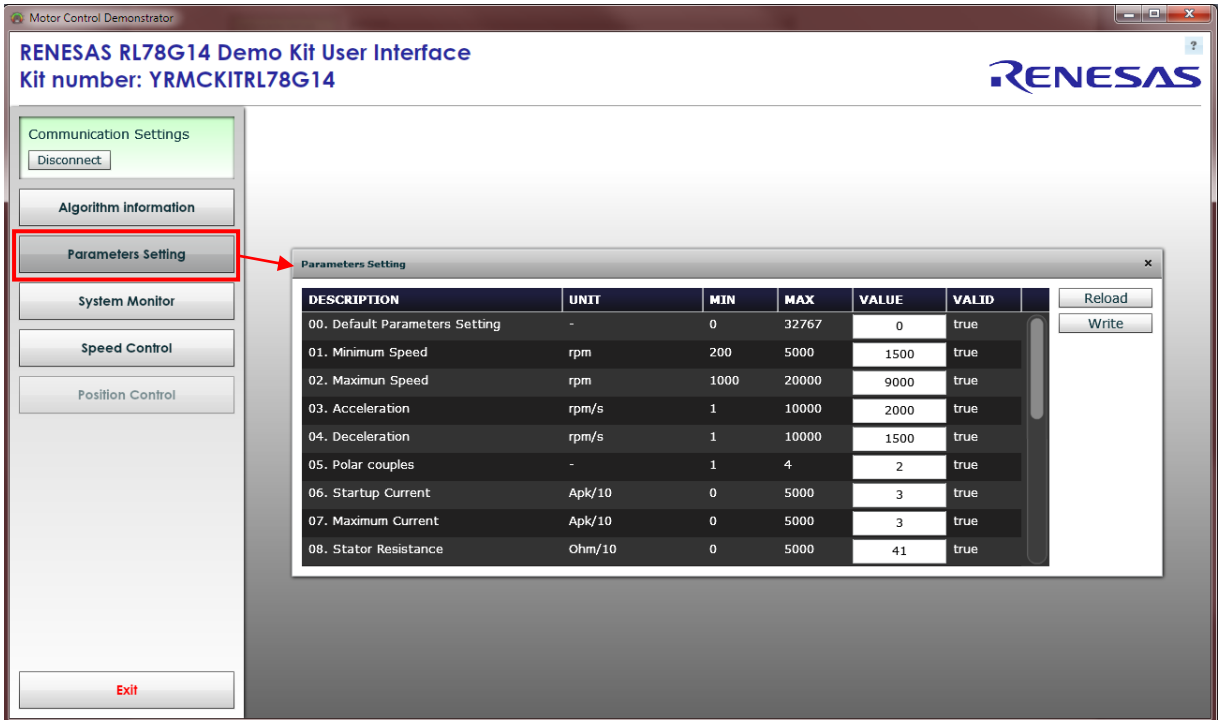
5.2.2. Parameter Setting

This window allows the run time parameters to be set and changed. When the parameter set is changed and re written the saves these in the non-volatile memory (Data Flash or EERPOM)
Before running the motor, check that the Parameters Settings displayed like the speed range and the number of polar couples etc. are in-line with the motor to be used or tuned (See section 12).

The parameters can be reloaded into the GUI by pressing the "Reload" button to read all the parameters that are stored in the non-volatile memory (eeprom or data flash).

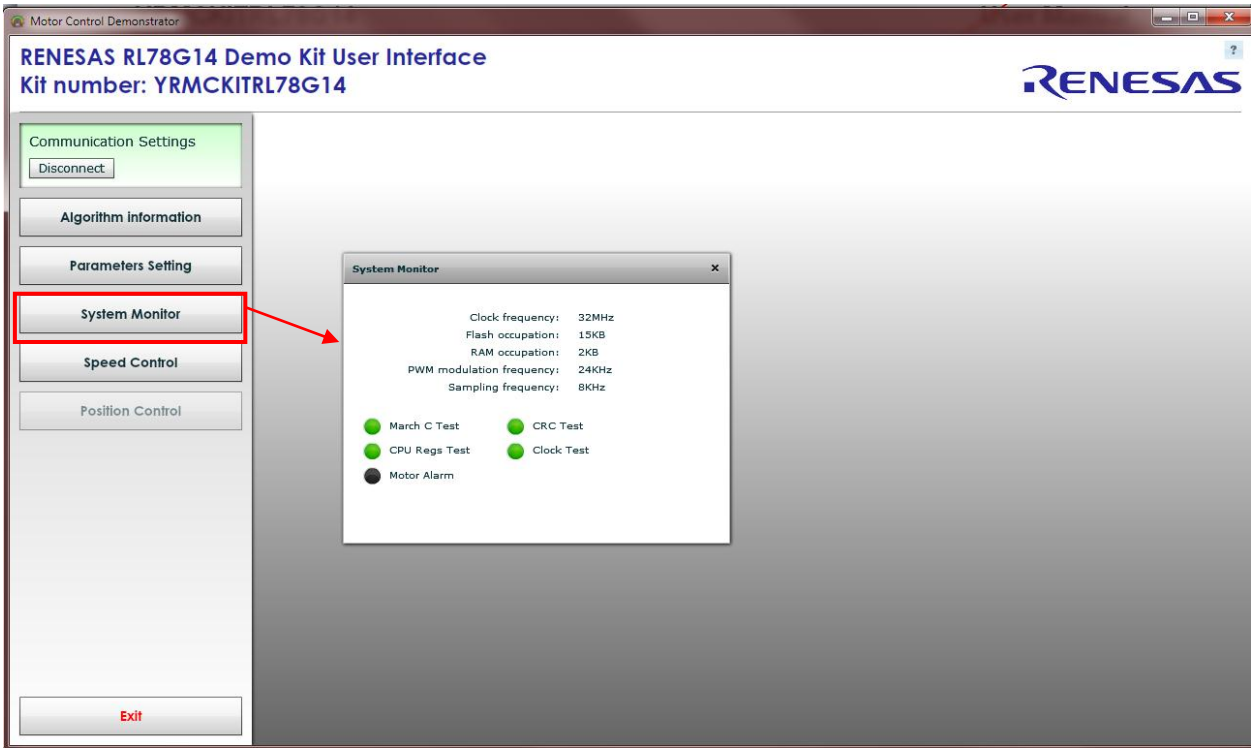
In case of setting incorrect or inconsistent parameters, the original default parameters can be restored from the MCU flash memory (Not Data Flash or EEPROM) by following the operation shown below

1. Enter the magic number "33" in the first line called: "00. Default Parameters setting"
2. Click the "Write" button in the parameter setting window
3. Then hardware RESET the RL78 by pressing the P6 button on the board.
4. Click the "Reload" button to get the default parameters defined in the "customize.h" header file in the IDE workspace.



5.2.3. System Monitor

This window shows the status of the RL78MCU self-test results and also shows if there is a motor error
The motor error/alarm is also shown in the main "Speed Control" Window



5.2.4. Main Control Window (Speed Control)

This window provides the main control window for the board

It includes the following features

a. RPM Control

Allows manual control of the motor including speed and direction.
The speed reference can be varied using the "RPM CONTROL" pointer, which can be either moved with the mouse, or with the value written, directly into the text box. A value written as a positive value (i.e. 3000) will operate the motor in a clockwise direction, with a value written as a negative value (i.e. -3000) will operate the motor in the reverse (Anti-clockwise) direction

"PAUSE", "GO" and "REVERSE" control buttons are also included.

The "DEMO" button enters the pre-set demonstration command sequence, which operates the motor with the same sequence as described previously for the "Stand Alone" demonstration. (i.e. the same operation as pressing the button P4 on the board). No other settings are required

The two bars shown in the RPM Control wheel indicate the "minimum speed" point as set in the parameter settings. Below the minimum speed the motor does not operate and above the minimum speed the motor operates normally at the reference speed set. The setting of the minimum speed can be displayed by the tool tip as the mouse is moved over the minimum indicator. This is the value set in the parameter settings window.

b. Property Monitor

This windows shows the main motor operating parameters

The monitor graph can be set to display any of monitored properties, by clicking on the required property to be viewed

The run time data can be saved to an Excel file by clicking on the “Save data to file” button
This will open a dialogue box to select the filename and location of the file that the data is saved into.
A control bar will appear to allow the use to Start, Pause or Stop the data recording process

c. Speed Control Graphs

There are three graphs that monitor the operation of the motor. SPEED, VOLTAGE and CURENT

The speed graph shows the Target and Measured Speed
The Voltage graph shows the VBUS Supply value, Direct, Quadrature and Total Voltages
The Current graph shows the Total, Direct and Total currents

Any individual graph can be expanded by pressing the “ZOOM” button. When a graph is “ZOOMED” then only this graph is visible in the GUI display. Data for all graphs continues to be updated in the background.

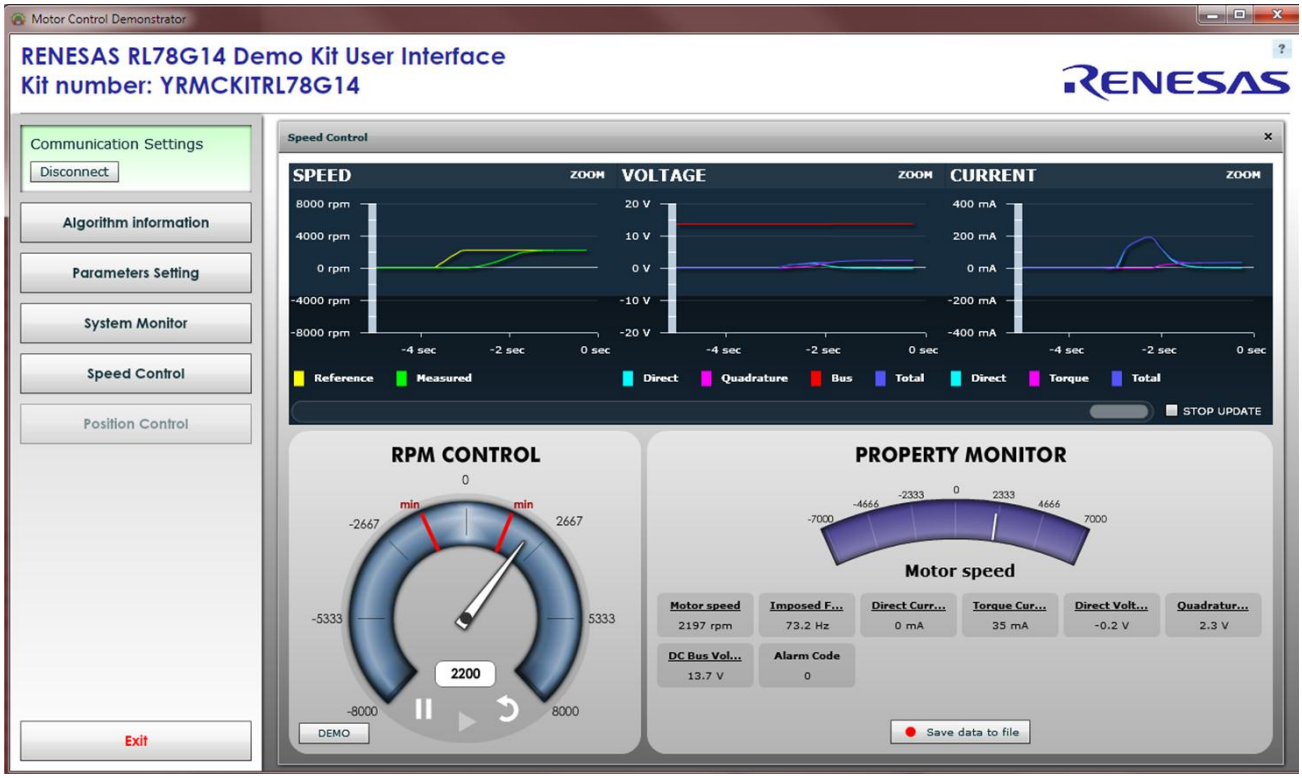
The previous data timeline of the graphs can be seen by using the slide control next to the “STOP UPDATE” button

The graphs update can be stopped by clicking on the STOP UPDATE

Once the GUI is connected, open the main control window by pressing the “Speed Control” button. Through this window, the user can control the speed reference to the board, and can view all the reference values and measurements. in the three “oscilloscope” graphs.

The motor can be controlled by the “RPM CONTROL” section by either moving the point to the desired speed and direction or setting the speed in the text box. The “DEMO” button performs the same operation as described previously.

Once the motor is in operation the graphs will start to update together with the measured values in the "PROERTY MONITOR" window as shown below



d. Alarm Codes

There are three "alarm codes used in the GUI and embedded software

1. EEPROM error alarm
2. Inverter error
3. Loss of phase error

The alarm codes usually can be cleared by resetting the motor speed back to zero. In some circumstances it is necessary to perform a manual reset of the RL78/G14 to clear the error. This may reset the connection between the board and the GUI.

e. Saving Monitor Property Data to a file

It is possible to save the measured data displayed in the "PROPERTY MONITOR" window to a "comma separated values` (csv) excel file for offline analysis and review

This is started by clicking the "Save Data to file" button located at the bottom of the "PROPERTY MONITOR" window.

This will open a media player style control bar which allows the user to

1. Stop recording , close the media bar and close the file
2. Pause / Resume recording
3. Reset Recording (Note this will overwrite the data in the file)

Each update of the GUI, monitor data is saved and will create a new entry in the excel file.

When the button is first pressed the user will be prompted for the file name of the file to be used. Please note that recording start as soon as the file is opened even if the motor is not yet running. This can be a new file or and existing file created previously. Note that a new session will overwrite any previous data in the file

Due to the nature of a “csv” file is may be possible on some versions of MS Office that the data is not correctly separated into columns. This can be corrected by changing the settings in the MS Office Excel program when the “csv” file is first opened so that it is saved correctly in future usage. Please refer to the Microsoft Office installation for further information.

5.3. GUI Control Command Flow

The following information shows the command and control flow for the communications between the PC GUI and the RL78/G14 motor control board firmware.

ASCII codes used as commands: “!” = 0x21, “#” = 0x23, “?” = 0x3F, “W” = 0x57, “c” = 0x63, “w” = 0x77

If the address "a" specified in the Master command is <NUM_PAR_EQP (number of eeprom parameters), then the parameter is read or written depending on the command.

Otherwise if the address “a” >=NUM_PAR_EQP, then a parameter in the ram table (userif.h) is read or written;

Its address (location) in the ram table is defined by “a” - NUM_PAR_EQP.

5.3.1. Master Control Codes

Frame Format

l i s o a n D1 .. Dm k where

l = frame total length (1 byte)

i = master string identifier ('?')

s = station address (1 byte)

o = operation code (1 byte)

a = data address (1 byte)

n = data number (1 byte)

Dx = x-th data byte (1 byte)

k = checksum (1 byte)

Master Command codes:

'**c**' = check

'**w**' = word reading (1 word = 2 bytes)

'**W**' = word writing (1 word = 2 bytes)

Possible master frames (Commands):

Check: **l ? s c k** (l=5)

Word read**: l ? s w a n k** (l=7)

Word write: **l ? s W a n D11 D10 .. Dn1 Dn0 k** (l=7+2*n)

5.3.2. Slave Control Codes

Slave string:

l i s o a n D1 .. Dm k where

l = frame total length (1 byte)

i = slave string identifier ('!' = OK answer, '#' = NOK answer)

s = station address (1 byte)

o = operation code (1 byte)

a = data address (1 byte)

n = data number (1 byte)

Dx = x-th data byte (1 byte)

k = checksum (1 byte)

Slave Command Code Operation:

'c' = check answer

'w' = word reading answer (word = 2 byte)

'W' = word writing answer (word = 2 byte)

Possible slave frames (Responses):

Nok: **l # s o k** (l=5)

Check: **l ! s c k** (l=5)

Word read: **l ! s w a n D11 D10 .. Dn1 Dn0 k** (l=7+2*n)

Word write: **l ! s W k** (l=5)

5.3.3. Communications Examples

Example 1

PC request of reading 16 words from the structure UIF_R, starting from the second position (UIF_R.ram_tab[1], .., UIF_R.ram_tab[16]):

Byte	Code	Meaning
0	07	Number of bytes in the frame
1	3F	Master command string indicator ("?")
2	00	Station address (it is always 0 in our boards)
3	77	word reading operation "w"
4	41	data start address (1(address in UIF_R.ram_tab) + 40h (offset to add for ram reading/writing))
5	10	number of data words (0x10, 16dec)
6	39	checksum

Board answer:

Byte	Code	Meaning
0	27	Number of bytes in the frame (27h=39dec)
1	21	Slave string indicator "!"
2	00	Station address (it is always 0 in our boards)
3	77	word reading operation "w"
4	41	data start address (1(address in UIF_R.ram_tab)+40h(offset to add for ram reading))
5	10	number of data (10h=16dec)
6	00	MSB of the 1st word of data (UIF_R.ram_tab[1]=UIF_R.var.rpm, speed)
7	00	LSB of the 1st word of data
8	00	MSB of the 2nd word of data (UIF_R.ram_tab[2]=UIF_R.var.fre, imposed frequency)
9	00	LSB of the 2nd word of data
10	00	MSB of the 3rd word of data (UIF_R.ram_tab[3]=UIF_R.var.id, d axis current)
11	00	LSB of the 3rd word of data
12	00	MSB of the 4th word of data (UIF_R.ram_tab[4]=UIF_R.var.iq, q axis current)
13	00	LSB of the 4th word of data
14	00	MSB of the 5th word of data (UIF_R.ram_tab[5])
15	00	LSB of the 5th word of data
16	00	MSB of the 6th word of data (UIF_R.ram_tab[6])
17	00	LSB of the 6th word of data
18	00	MSB of the 7th word of data (UIF_R.ram_tab[7]=UIF_R.var.vb, bus voltage)
19	18	LSB of the 7th word of data
20	00	MSB of the 8th word of data (UIF_R.ram_tab[8])
21	00	LSB of the 8th word of data
22	00	MSB of the 9th word of data (UIF_R.ram_tab[9]=UIF_R.var.all, alarm)
23	01	LSB of the 9th word of data
24	00	MSB of the 10th word of data (UIF_R.ram_tab[10])
25	00	LSB of the 10th word of data
26	00	MSB of the 11th word of data (UIF_R.ram_tab[11])
27	00	LSB of the 11th word of data
28	00	MSB of the 12th word of data (UIF_R.ram_tab[12])
29	00	LSB of the 12th word of data
30	00	MSB of the 13th word of data (UIF_R.ram_tab[13])
31	00	LSB of the 13th word of data
32	00	MSB of the 14th word of data (UIF_R.ram_tab[14])
33	00	LSB of the 14th word of data
34	00	MSB of the 15th word of data (UIF_R.ram_tab[15])
35	00	LSB of the 15th word of data
36	00	MSB of the 16th word of data (UIF_R.ram_tab[16])
37	00	LSB of the 16th word of data
38	69	checksum

Example 2

PC request of writing 4 words in the structure UIF_W, starting from the third position (UIF_W.ram_tab[2], .., UIF_W.ram_tab[5]):

Byte	Code	Meaning
0	0F	Number of bytes in the frame (0Fh=15dec)
1	3F	Master string indicator. ("?")
2	00	Station address (it is always 0 in our boards)
3	57	word writing operation "W"
4	42	data start address (2(address in UIF_W.ram_tab)+40h(offset to add for ram reading/writing))
5	04	number of data
6	03	MSB of the first word of data (value (03E8h=1000dec) to be written in UIF_W.ram_tab[2] = UIF_W.var.rif (speed reference)
7	E8	LSB of the first word of data
8	00	MSB of the second word of data (value to be written in UIF_W.ram_tab[3], not used)
9	00	LSB of the second word of data
10	00	MSB of the third word of data (value to be written in UIF_W.ram_tab[4], not used)
11	00	LSB of the third word of data
12	00	MSB of the fourth word of data (value to be written in UIF_W.ram_tab[5], not used)
13	00	LSB of the fourth word of data
14	E7	checksum

Board answer (indicates that the request is received and processed):

Byte	Code	Meaning
0	05	Number of bytes in the frame
1	21	Slave string indicator "!"
2	00	Station address (it is always 0 in our boards)
3	57	word writing operation "W"
4	E6	checksum

6. Motor Calibration using the PC GUI Interface

A full calibration of the motor can be performed via the PC User Interface.
The most important parameters to test are the following:

1. Stator resistance
2. Start-up current
3. Maximum current
4. Synchronous inductance
5. Current PI parameters
6. Speed PI parameters

Warning:

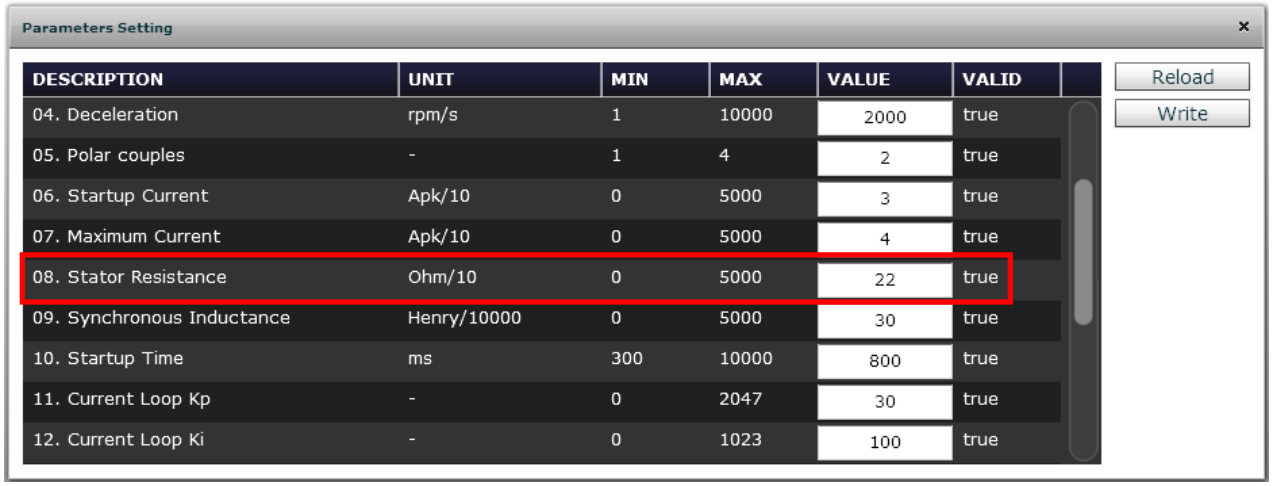
Do not try to start the motor before entering the initial calibration parameters otherwise the system could be damaged.

6.1. Stator Phase Resistance

The Phase resistance can be found by measuring the phase-to-phase resistance using a meter.
The measured value should be divided by 2.
Please enter the value in the parameter 08, "Stator Resistance" as shown below.

Click the button "Write" to save the new parameter value.

Since the measurement unit is Ohm/10, the value 23 of the picture means 2.3 Ohm.

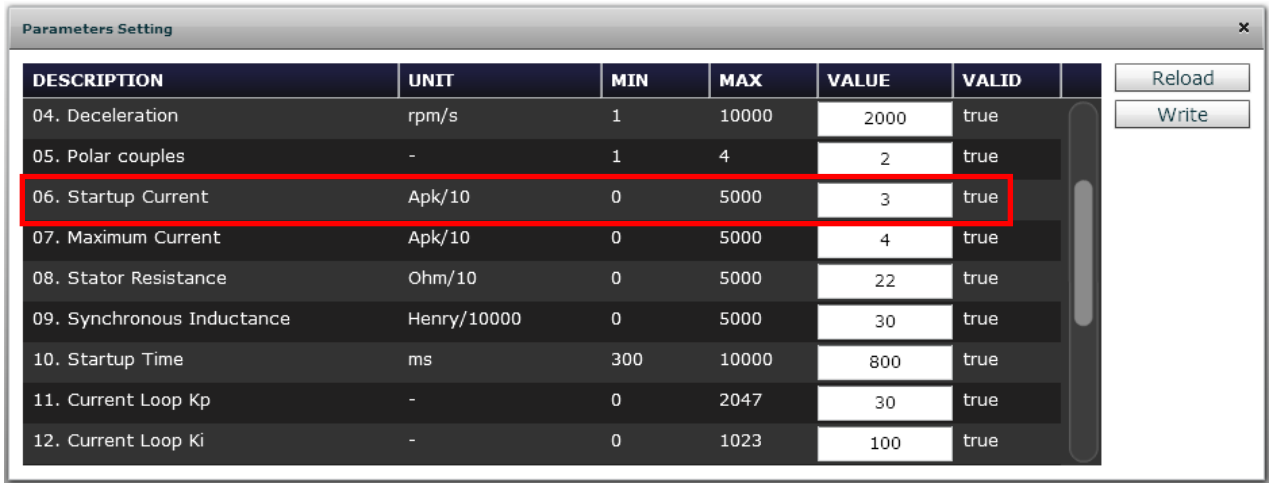


6.2. Start Up current

The start-up current parameter is responsible for the proper motor start-up. Please enter an average value that will not damage the motor. This value can be increased slowly if the motor fails to start at the end of the procedure.

Click the button “Write” to save the new parameter value.

Note: The value 3 in the picture below means 0.3 Amperes.

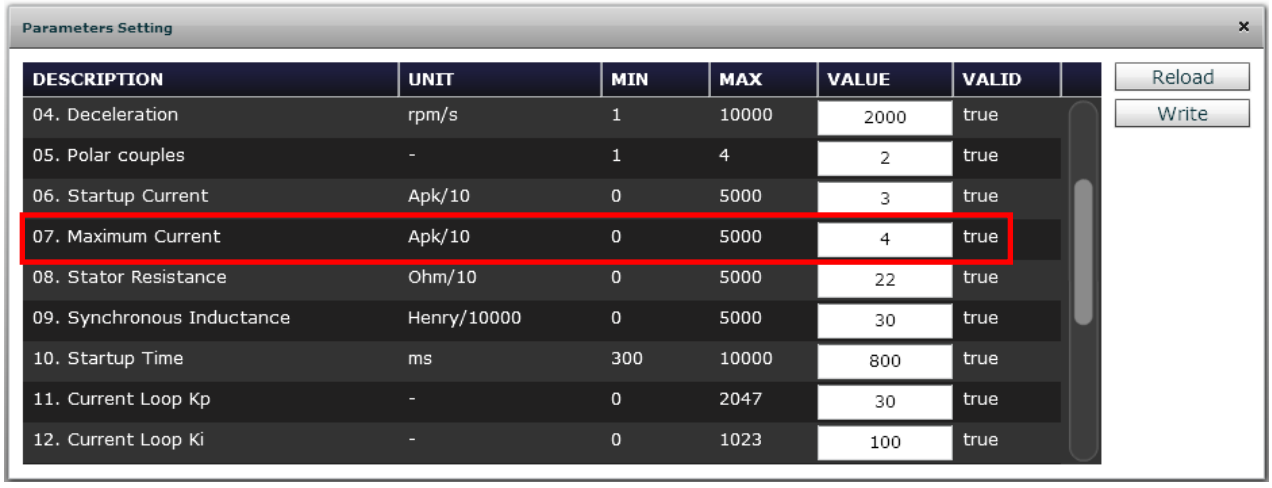


Setting the start-up current during motor tuning

6.3. Maximum Current

Maximum current specifies the maximum current to be imposed.
It should be set accordingly the inverter and the motor specifications, and as required by the application.
Click the button “Write” to save the new parameter value.

The value 3 in the picture below means 0.3 Amperes.



Setting the maximum current for motor tuning

6.4. Synchronous Inductance

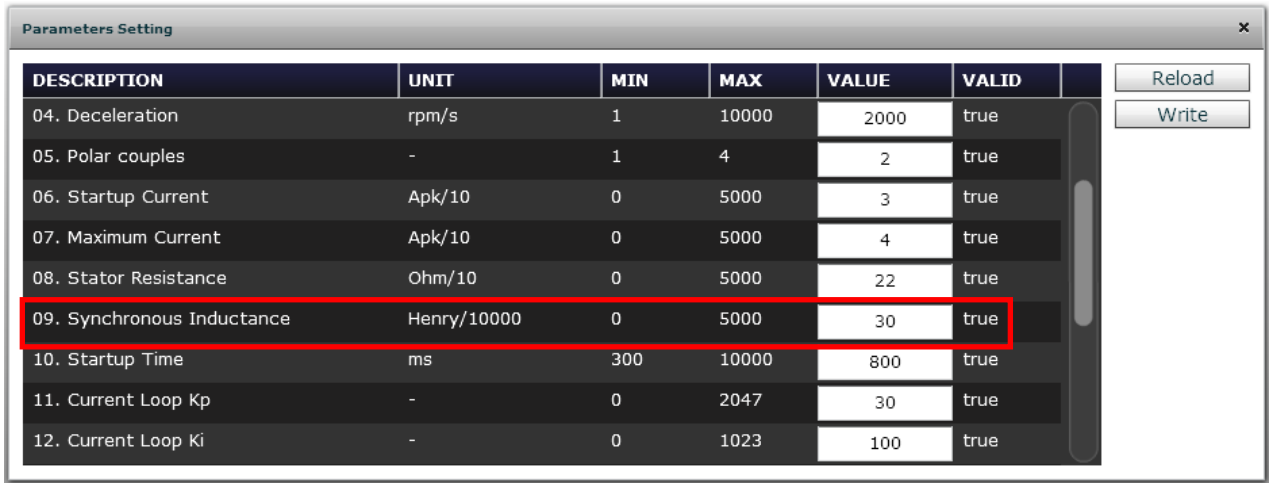
Synchronous Inductance represents the average synchronous inductance of the motor (it is an equivalent quantity that represents the auto and mutual interactions between the phase currents); it is usually low in surface magnets motors; it can be neglected for the first tuning of the motor, but a wrong value will introduce a phase error proportional to the load, so the tuning strategies can be either:

- Maximize the torque at a rated current;
- Minimize the current at a rated torque.

Please enter the value **zero** for the first calibration as shown below (the first start-up has to be done with no-load; in this case there is no phase error neglecting the inductance).

Click the button "Write" to save the new parameter value.

Since the measurement unit is Henry/10000, the value 100, for example, means 10mH (milliHenry)

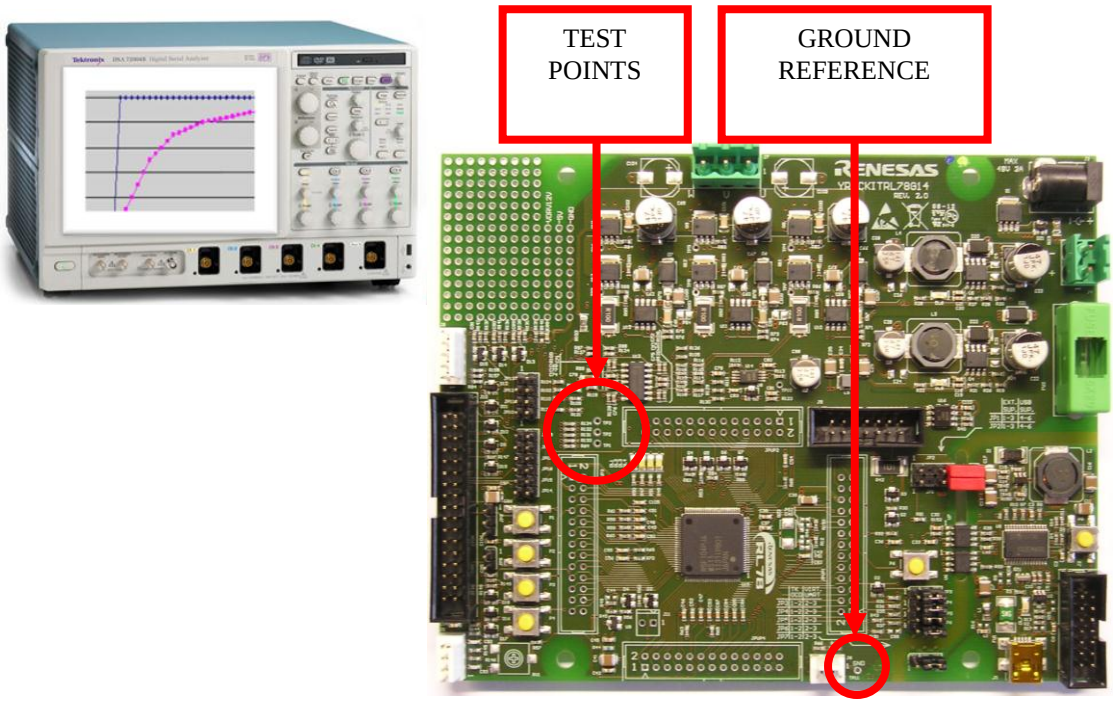


Setting the initial Synchronous Inductance during tuning

6.5. Tuning the initial Current Pi Gains

Current PI parameters allow a proper current control of the motor and the system.

The software offers a particular procedure to help the tuning of the current PI gains. An oscilloscope is needed to see the response of the system to the stimulation; the figure below is showing the Test Point TP1 to be used for the calibration.



Current Pi Gain Setting – Monitor point Signal Connections

First of all it is necessary to enable the "current PI tuning mode", and this can be done entering the magic value "22" in the first parameter ("Default parameters setting") in the "setup" and click on the button "Write".

Now perform a hardware reset on the RL78G14, so the calibration procedure of the Current PI coefficient is enabled.

In the parameters list, please set the current PI coefficients: Kp and Ki to "1" as shown below:

Parameters Setting

DESCRIPTION	UNIT	MIN	MAX	VALUE	VALID
08. Stator Resistance	Ohm/10	0	5000	22	true
09. Synchronous Inductance	Henry/10000	0	5000	30	true
10. Startup Time	ms	300	10000	800	true
11. Current Loop Kp	-	0	2047	1	true
12. Current Loop Ki	-	0	1023	1	true
13. Speed Loop Kp	-	0	4095	50	true
14. Speed Loop Ki	-	0	4095	200	true
15. Free	-	0	65535	0	true
16. Free	-	0	32767	0	true

Reload
Write

Initial Setting of the Current Kp and Ki gains for motor tuning

Connect the oscilloscope probes to the Test Point (TP1) and to the Ground of the board.

The next step is to generate a current step and capture the current PI controller response in order to tune the current PI coefficients. To generate the current step and check the response on the oscilloscope, please enter "1" in the parameter 17, "PI Tuning trigger" and click on the button "Write". This will generate a current step reference which amplitude is equal to the value specified as start-up current. In the oscilloscope you will observe the internal measurement of the obtained current, normalized in such a way that when the signal is equal to 4V the current is equal to the requested one. So the tuning procedure consists in ***varying the PI gains in order to make the signal equal to 4V in the fastest possible way***, without oscillation.

Please find below in the picture some examples of the different responses you may obtain.

Test Point 1 (TP1) Oscilloscope Tuning Outputs

Keep in mind that the proportional gain is the responsible of the "reactivity" of the system, while the integral gain allows reducing to zero the steady state error. An excess of proportional gain will produce high frequency oscillations (usually with audible noise), while an excess of integral action will produce lower frequency oscillations. Note also that if the gains are too low, no answer is produced; increasing the parameters, at a certain level, the current obtained will be enough to move the rotor, which will "align"; from this point in advance, don't move the rotor by hand, and you will see valid answers in the oscilloscope. The final rise time will depend on the motor you are using, but usually a rise time lesser than 1ms can be obtained.

After tuning the current PI parameters manually reset the board in order to return to the "normal" operation mode.

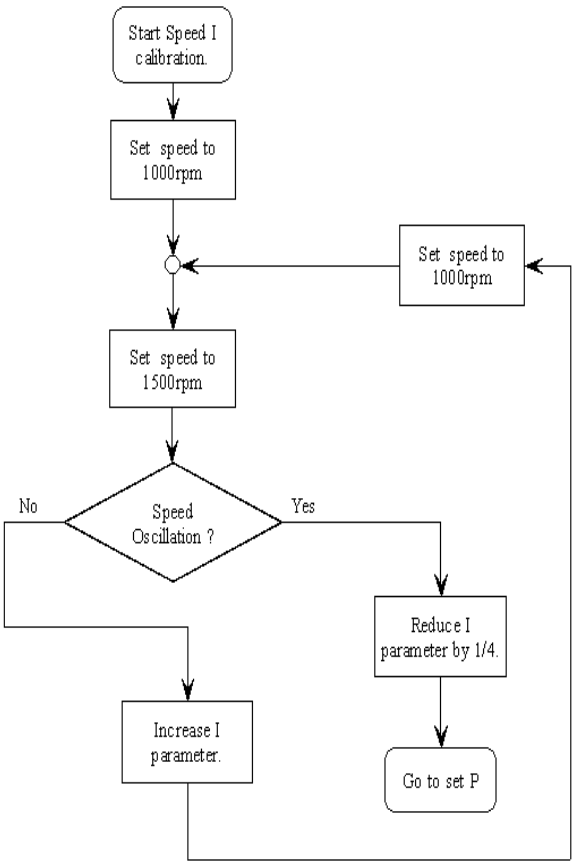
6.6. Tuning the Speed Pi Parameters

Speed PI proportional gain (par. 13, “Speed Loop Kp”) and integral gain (par. 14, “Speed Loop Ki”) should be tuned in the real application and under load conditions.
As starting values, low values can be chosen; they can be increased at medium working speed until instability arises; (high frequency instability is related to the proportional value too high, low frequency instability is related to integral value too high). When instability arises, the value should be halved. Some kind of tuning of speed parameters can be performed using high values of acceleration ramp, and imposing speed reference variations, as done with the current PIs.

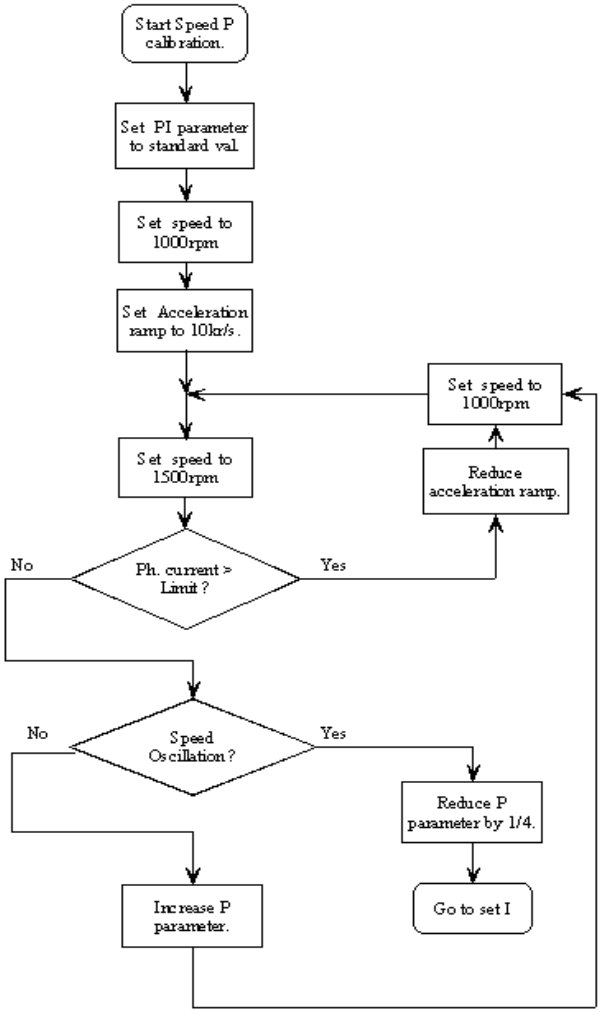
The PI calibration procedure should be iterated till the desired system response is reached. The speed reference could be changed depending on the motor/application. You can find below two graphs indicating an example of tuning procedure; this procedure should be made using the real working environment.

Speed parameters can influence the success of the start-up phase: if the algorithm fails in this phase, giving alarm n°3, try modifying the speed proportional gain first, and then the integral gain.

Speed Pi Integral calibration Flow Chart



Speed Pi Proportional calibration Flow Chart

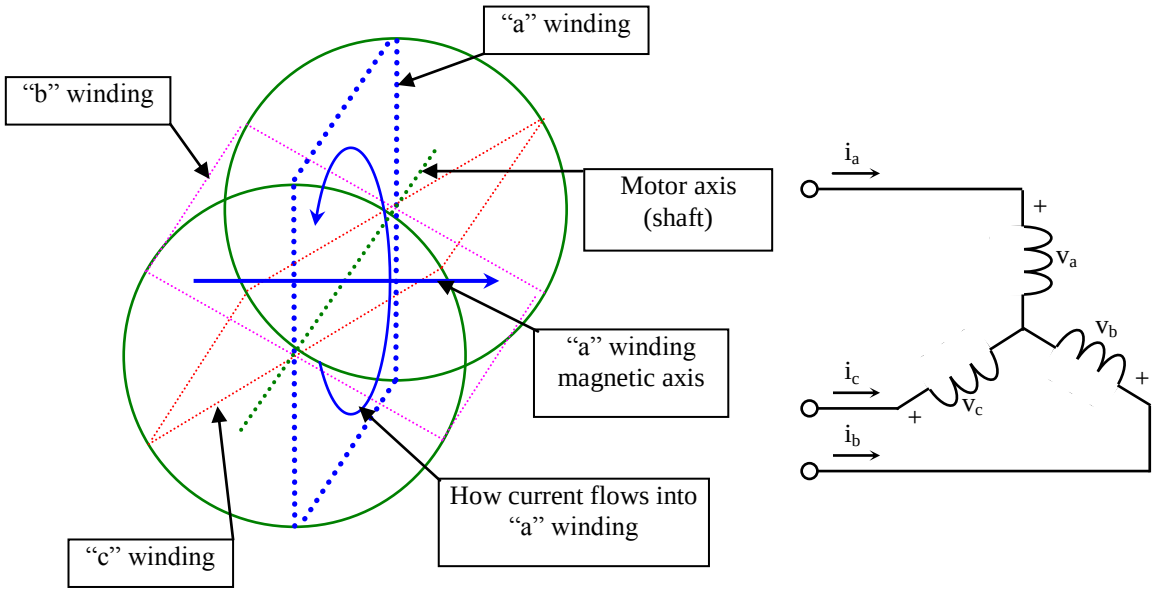


7. Permanent magnets brushless motor model

The synchronous permanent magnets motor (sinusoidal brushless motor) is widely used in the industry. More and more home appliance makers are now using such brushless motor, mainly because of the intrinsic motor efficiency.

The permanent magnet motor is made with few components:

- A stator formed by stacking sheared metal plates where internally the copper wiring is wound, constructing the stator winding.
- A rotor in which permanent magnets are fixed.
- Two covers with ball bearings that keep together the stator and the rotor; the rotor is free to rotate inside the stator.



PMAC Motor Model

The working principle is quite simple: if we supply the motor with a three-phase system of sinusoidal voltages, at constant frequency, in the stator windings flow sinusoidal currents, which create a rotating magnetic field.

The permanent magnets in the rotor tend to stay aligned with the rotating field, so the rotor rotates at synchronous speed.

The main challenge in driving this type of motor is to know the rotor position in real-time, so mainly implementation are using a position sensor or a speed sensor.

In our implementation, the system is using either one or three shunts to detect the rotor position in real-time.

Let's analyse the motor from a mathematic point of view.

If we apply three voltages va(t), vb(t), vc(t) to the stator windings, the relations between phase voltages and currents are:

$$v_a = R_S i_a + \frac{d\lambda_a}{dt}$$

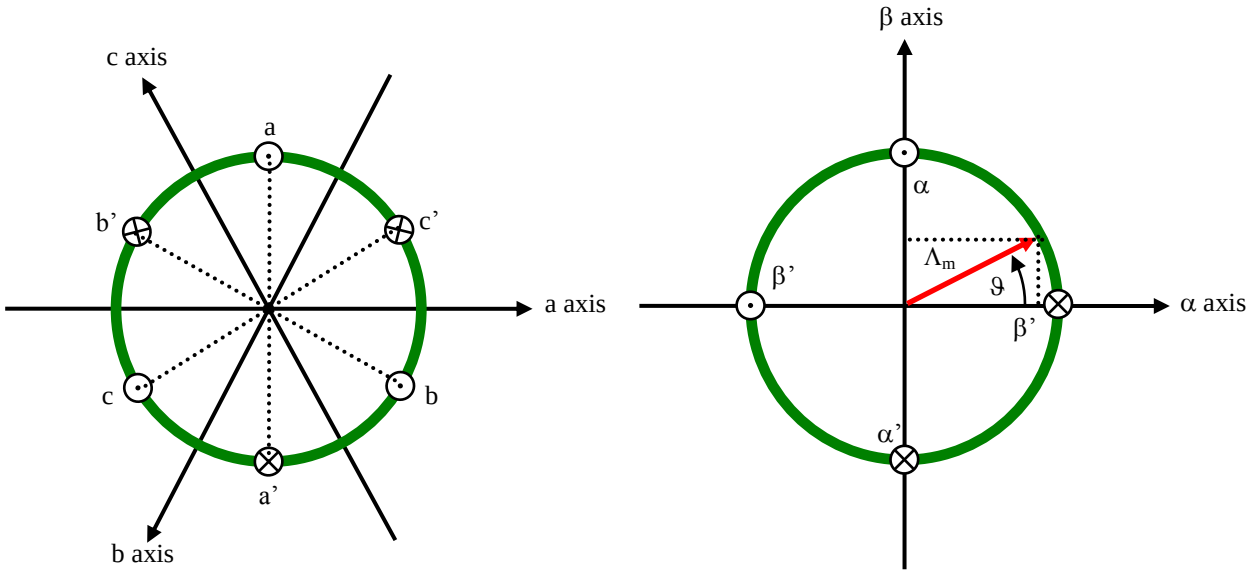
$$v_b = R_S i_b + \frac{d\lambda_b}{dt}$$

$$v_c = R_S i_c + \frac{d\lambda_c}{dt}$$

- λ_i is the magnetic flux linkage with the i-th stator winding.

- R_S is the stator phase resistance (the resistance of one of the stator windings).

The magnetic flux linkages λ_i are composed by two items, one due to the stator currents, one to the permanent magnets.



Real axes (a, b, c) and equivalent ones (α, β); a fixed amplitude vector can be completely determined by its position respect the (α, β) system (angle ϑ)

The permanent magnet creates a magnetic field that is constant in amplitude and fixed in position in respect to the rotor. This magnetic field can be represented by vector Λ_m whose position in respect to the stator is determined by the angle ϑ between the vector direction and the stator reference frame.

The contribution of the permanent magnets in the flux linkages depends on the relative position of the rotor and the stator represented by the mechanical-electric angle ϑ.

It is, in every axis, the projection of the constant flux vector Λ_m in the direction of the axis:

$$\lambda_a = Li_a + \Lambda_m \cos(\vartheta)$$
$$\lambda_b = Li_b + \Lambda_m \cos(\vartheta - 2\pi/3)$$
$$\lambda_c = Li_c + \Lambda_m \cos(\vartheta - 4\pi/3)$$

Supposing that the rotor is rotating at constant speed ω (that is: ϑ(t) = ωt) the flux linkages derivatives can be calculated, and we obtain:

$$v_a = R_S i_a + L\frac{di_a}{dt} - \omega\Lambda_m \sin(\vartheta)$$
$$v_b = R_S i_b + L\frac{di_b}{dt} - \omega\Lambda_m \sin(\vartheta - 2\pi/3)$$
$$v_c = R_S i_b + L\frac{di_b}{dt} - \omega\Lambda_m \sin(\vartheta - 4\pi/3)$$

A "three phase system", may be represented by an equivalent "two phase system". So there by using specific transformations, our three equations system is equivalent to a two equations system. It is basically a mathematical representation in a new reference coordinates system.

In the two phases (α,β) fixed system the above equations become:

$$v_\alpha = R_S i_\alpha + \frac{d\lambda_\alpha}{dt}$$
$$v_\beta = R_S i_\beta + \frac{d\lambda_\beta}{dt}$$

For the magnetic field equations, we got:

$$\lambda_\alpha = Li_\alpha + \lambda_{\alpha m} = Li_\alpha + \Lambda_m \cos(\vartheta)$$
$$\lambda_\beta = Li_\beta + \lambda_{\beta m} = Li_\beta + \Lambda_m \sin(\vartheta)$$

After performing the derivation:

$$\frac{d\lambda_\alpha}{dt} = L\frac{di_\alpha}{dt} - \omega\Lambda_m \sin(\vartheta) = L\frac{di_\alpha}{dt} - \omega\lambda_{\beta m}$$
$$\frac{d\lambda_\beta}{dt} = L\frac{di_\beta}{dt} + \omega\Lambda_m \cos(\vartheta) = L\frac{di_\beta}{dt} + \omega\lambda_{\alpha m}$$

Finally, we obtain for the voltages in (α,β) system:

$$v_\alpha = R_S i_\alpha + L\frac{di_\alpha}{dt} - \omega\lambda_{\beta m}$$
$$v_\beta = R_S i_\beta + L\frac{di_\beta}{dt} + \omega\lambda_{\alpha m}$$

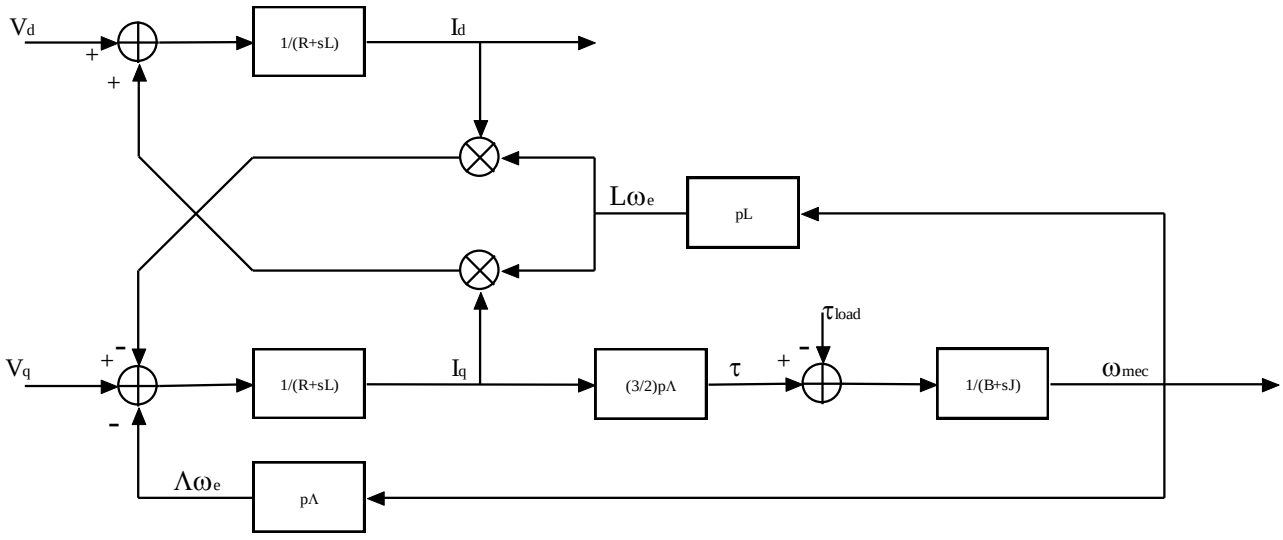
A second reference frame is used to represent the equations as the frame is turning at the rotor speed. So the "d" axis is chosen in the direction of the magnetic vector Λ_m, and with the "q" axis orthogonal to the "d" axis. The new reference system is (d, q).

The reference frame transformations from the (α,β) system to the (d, q) system depends on the instantaneous position angle ϑ.

So we obtain two inter-dependant equations in the (d, q) system:

$$v_d = R_S i_d + L\frac{di_d}{dt} - \omega L i_q$$
$$v_q = R_S i_q + L\frac{di_q}{dt} + \omega L i_d + \omega\Lambda_m$$

These two equations represent the mathematical motor model.



Vd and Vq Equation Diagram

A control algorithm which wants to produce determined currents in the (d, q) system must impose voltages given from the formulas above. This is ensured by closed loop PI control on both axis "d" and "q" (Proportional Integral).

Since there is a mutual influence between the two axes, decoupling terms can be used.

In the block scheme the mechanic part is included, where “p” is the number of pole pairs, while “B” represents friction, “J” the inertia, “τ_{load}“ the load torque and “τ” the motor torque:

$$\tau = \frac{3}{2} \times p \times \Lambda$$

The angular speed ω is represented in the scheme as ω_e to distinguish the electrical speed from the mechanical one.

Let’s now consider the equations we have seen in (α,β) system:

$$v_\alpha = R_S i_\alpha + \frac{d\lambda_\alpha}{dt}$$
$$v_\beta = R_S i_\beta + \frac{d\lambda_\beta}{dt}$$

These equations show that magnetic flux can be obtained from applied voltages and measured currents simply by integration:

$$\lambda_\alpha = \lambda_{\alpha 0} + \int_0^t (v_\alpha - R_S i_\alpha) dt$$
$$\lambda_\beta = \lambda_{\beta 0} + \int_0^t (v_\beta - R_S i_\beta) d$$

Furthermore:

$$\Lambda_m \cos(\vartheta) = \lambda_\alpha - L i_\alpha$$
$$\Lambda_m \sin(\vartheta) = \lambda_\beta - L i_\beta$$

If the synchronous inductance L is small, the current terms can be neglected, if not they have to be considered. In general:

$$x = \Lambda_m \cos(\vartheta) = \lambda_\alpha - L i_\alpha = \lambda_{\alpha 0} + \int_0^t (v_\alpha - R_S i_\alpha) dt - L i_\alpha$$
$$y = \Lambda_m \sin(\vartheta) = \lambda_\beta - L i_\beta = \lambda_{\beta 0} + \int_0^t (v_\beta - R_S i_\beta) dt - L i_\beta$$

So in the (α,β) system phase we obtain from the flux components:

$$\vartheta = \arctan(x / y)$$

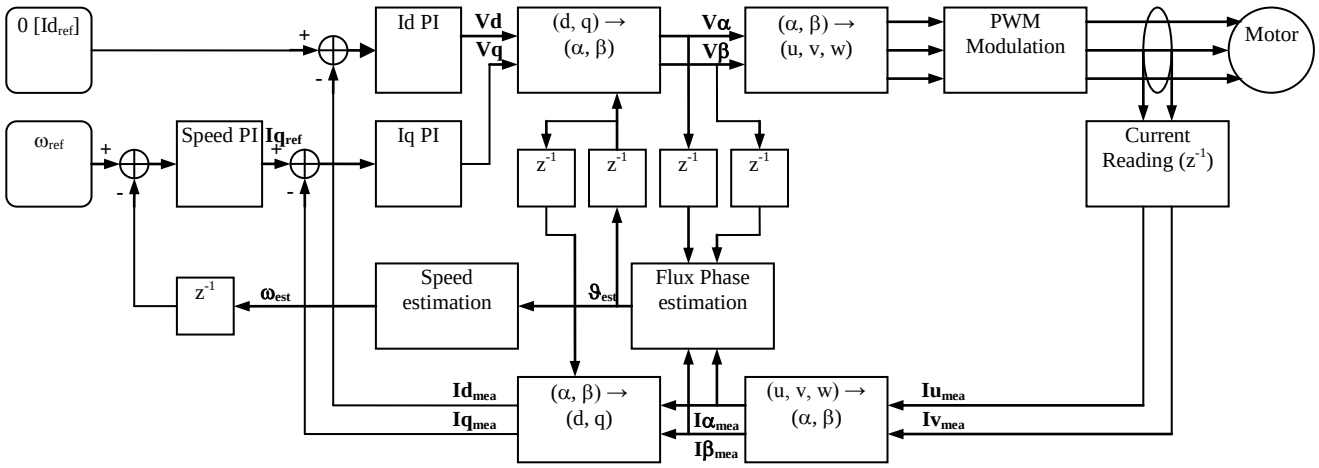
The system speed ω can be obtained as the derivative of the angle ϑ.

$$\omega = \frac{d}{dt} \vartheta(t)$$

Based on this, a sensor-less control algorithm was developed to give the imposed phase voltages, to measure phase currents, to estimate the angular position ϑ and finally the system speed.

8. Sensorless F.O.C. algorithm

Please, find below the sensor-less FOC algorithm block diagram.



Sensorless F.O.C algorithm

9. Software Description

In the YRMCKITRL78G14 kit the software is working on an RL78G14 [32MHz].

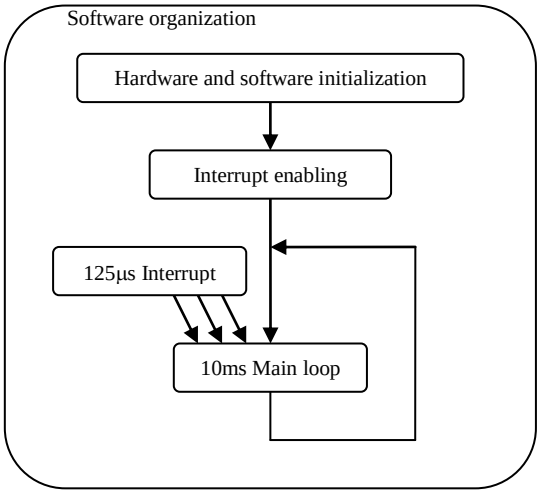
The total software uses the following resources:

1) FLASH 15Kbytes

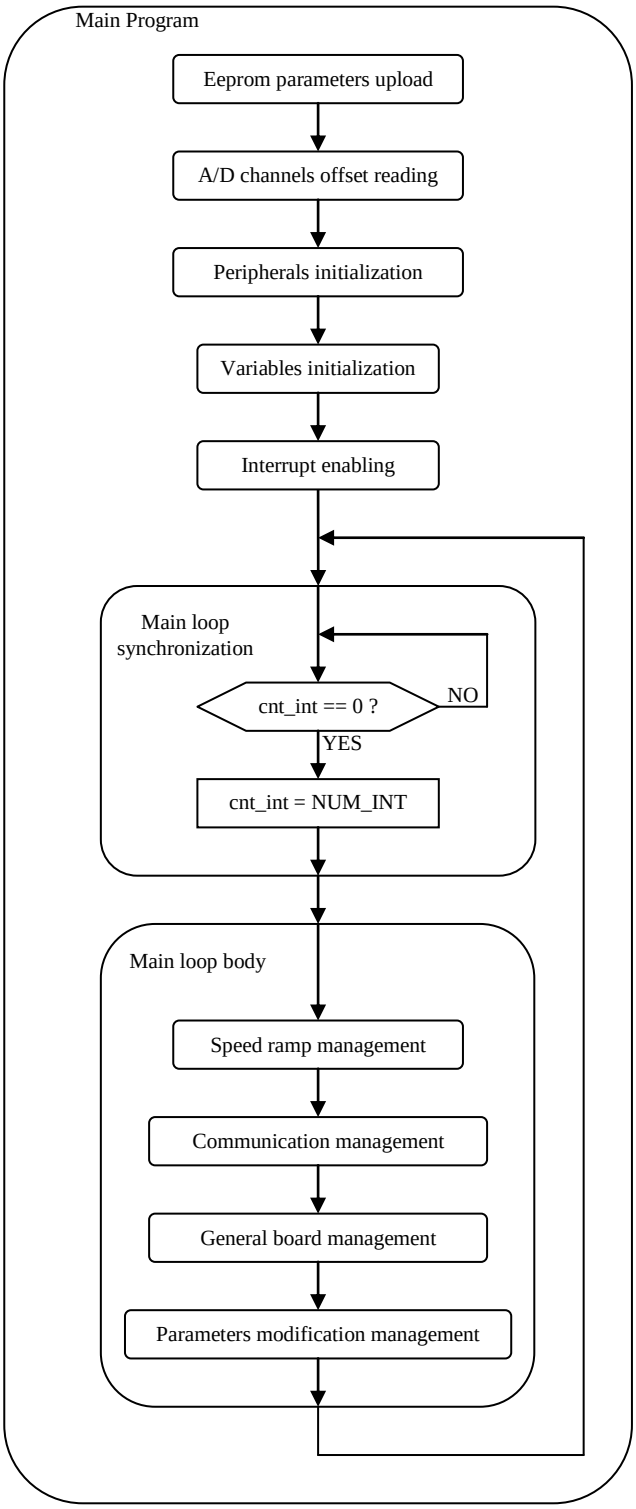
2) RAM : 2Kbytes

Please Note that this data also include the communication interface and the demo board management.

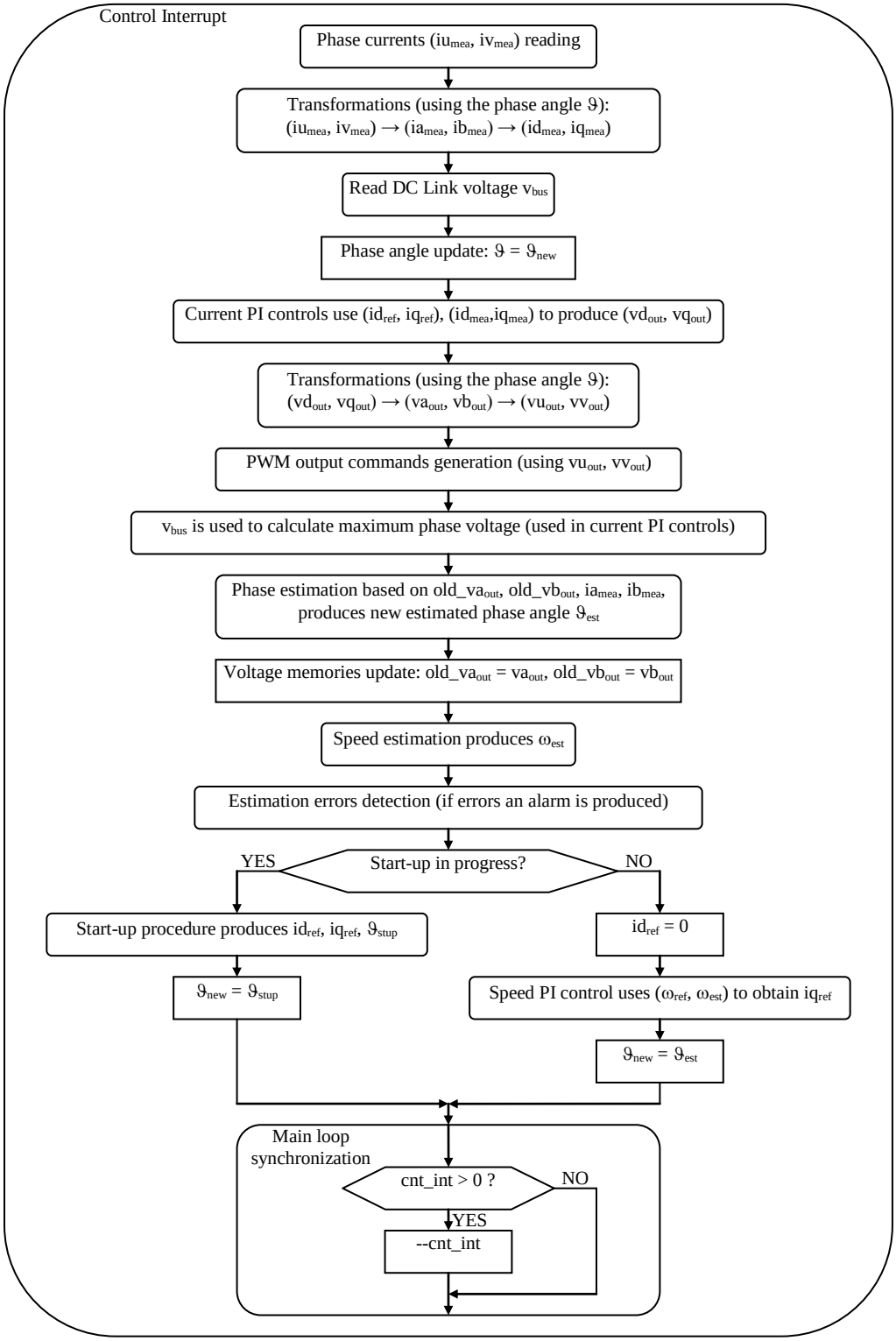
The following flow charts show the software implementation of the motor control part of the software



Software Structure



Main Control Flow



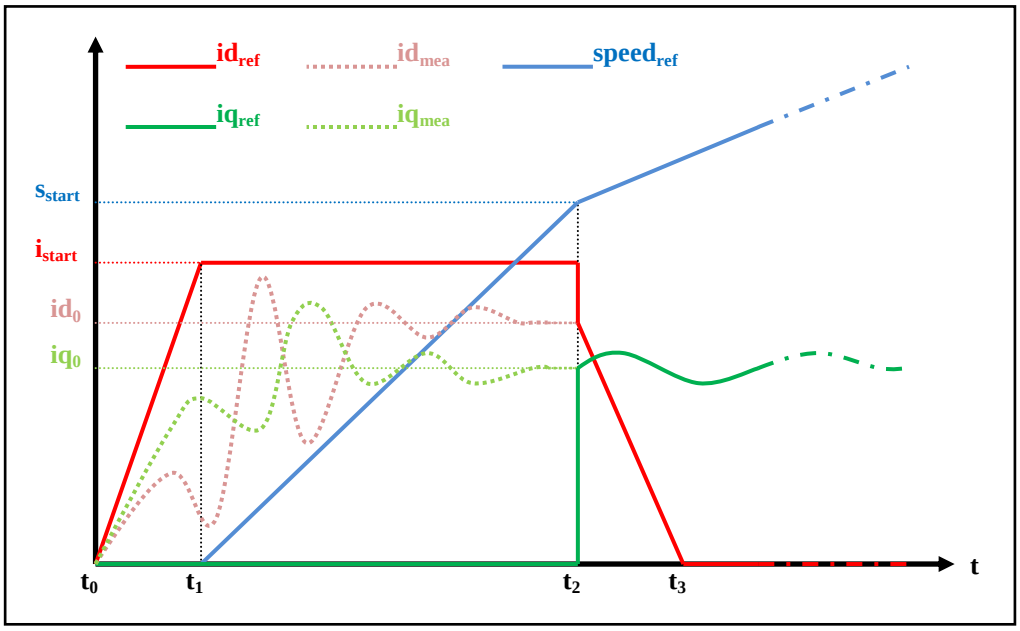
Motor Control Interrupt Processing

10. Start-up Procedure

When the motor is in stand-still, the phase of the permanent magnet flux vector cannot be detected with the used algorithm. So an appropriate start-up procedure has to be applied.

The idea is to move the motor in feed-forward (with higher current than that required to win the load), till a speed at which the estimation algorithm can work. Then the system can be aligned to the estimated phase, and the current can be reduced to the strictly necessary quantity.

The following graph illustrates the strategy used (the suffix "$_{ref}$" stands for reference, the suffix "$_{mea}$" stands for measured).



Start Up Process Graph

Referring to the graph, the start-up procedure (in case of three shunts current reading) is described below.

At the beginning t_0, the system phase is unknown. No current is imposed to the motor; the system phase is arbitrarily decided to be $\vartheta_a = 0$. All the references: id_{ref}, iq_{ref} and $speed_{ref}$ are set to zero.

From the moment t_0, while the iq_{ref} and the $speed_{ref}$ are maintained to zero, id_{ref} is increased with a ramp till the value i_{start} is reached at the moment t_1.

The references are referred to an arbitrary (d_a, q_a) system based on the arbitrary phase ϑ_a. From this moment, the phase estimation algorithm begins to be performed, and the estimated phase is ϑ_{est}.

The components of the current referred to the arbitrary (d_a, q_a) system are controlled to follow the references by the current PI controllers, so they will be $i_d=i_{start}$ and $i_q=0$. If we refer the measurements to the estimated phase we would obtain those which in the graph are called id_{mea} and iq_{mea} (referred to (d, q) estimated system); since the phase ϑ_{est} is still not correctly estimated, id_{mea} and iq_{mea} have no physical meaning.

At $t = t_1$, while iq_{ref} is maintained to zero and id_{ref} is maintained to its value i_{start}, $speed_{ref}$ is increased with a ramp till the value s_{start} is reached at $t = t_2$. The system phase $\vartheta_a(t)$ is obtained simply by integration of $speed_{ref}$; in the meanwhile, the phase estimation algorithm begins to align with the real system phase. Furthermore id_{mea} and iq_{mea} begin to be similar to the real flux and torque components of the current.

The interval (t_2-t_1) is the start-up time, and it is supposed to be large enough to allow the estimation algorithm to reach the complete alignment with the real phase of the system.

At $t = t_2$, the phase estimation process is supposed to be aligned. At this point a reference system change is performed: from the arbitrary (d_a, q_a) reference to the (d, q) reference based on the estimated phase ϑ_{est}.

The system reference change is performed as follows:

- The current references in the (d_a, q_a) system, $id_{ref}=i_{start}$ and $iq_{ref}=0$, are projected in the fixed system (α, β), to compute the instantaneous current components; the same is done with the integral parts of the current PI controllers, which are the mean voltages required to obtain those currents; in this way we obtain i_α, i_β, v_α, v_β.
- The phase is updated to ϑ_{est}.
- The α and β components obtained before are projected into the new reference system (d, q), giving the new current reference values i_{d0}, i_{q0} and the new PI integral memories v_{d0}, v_{q0}; the speed PI integral memory is loaded with the q current reference.

After $t > t_2$, the normal control is performed, based on the estimated phase ϑ_{est}; the speed reference is increased with the classical ramp; the i_d current reference is decreased with a ramp, till it reaches the value zero at the moment t_3; then it is maintained to zero; the i_q current reference is obtained as output of the speed PI controller.

11. Reference System Transformations in Detail

Find below the detailed equations used for the coordinates transformations.

(u, v, w) → (α, β):

$$g_\alpha = \frac{2}{3}(g_u - \frac{1}{2}g_v - \frac{1}{2}g_w) = g_a$$
$$g_\beta = \frac{2}{3}(\frac{\sqrt{3}}{2}g_v - \frac{\sqrt{3}}{2}g_w) = \frac{1}{\sqrt{3}}(g_v - g_w) = \frac{1}{\sqrt{3}}(g_u + 2g_v)$$

(α, β) → (u, v, w):

$$g_u = g_\alpha$$
$$g_v = -\frac{1}{2}g_\alpha + \frac{\sqrt{3}}{2}g_\beta = (-g_\alpha + \sqrt{3}g_\beta)/2$$
$$g_w = -\frac{1}{2}g_\alpha - \frac{\sqrt{3}}{2}g_\beta = (-g_\alpha - \sqrt{3}g_\beta)/2$$

(α, β) → (d, q):

$$g_d = g_\alpha \cos(\vartheta) + g_\beta \sin(\vartheta)$$
$$g_q = -g_\alpha \sin(\vartheta) + g_\beta \cos(\vartheta)$$
$$\vartheta = \omega t$$

(d, q) → (α, β):

$$g_\alpha = g_d \cos(\vartheta) - g_q \sin(\vartheta)$$
$$g_\beta = g_d \sin(\vartheta) + g_q \cos(\vartheta)$$
$$\vartheta = \omega t$$

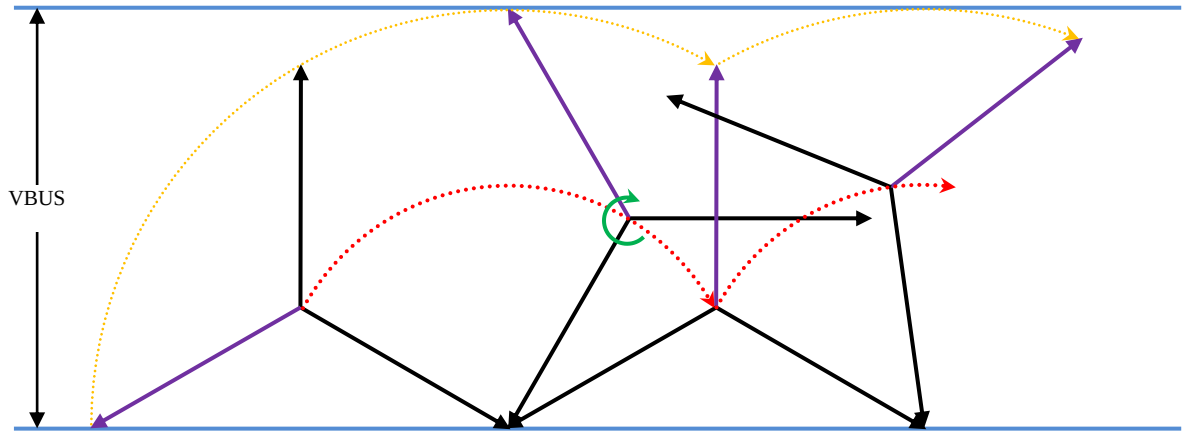
$$\begin{Bmatrix} g_u = G\cos(\omega t + \varphi_0) \\ g_v = G\cos(\omega t + \varphi_0 - 2\pi/3) \\ g_w = G\cos(\omega t + \varphi_0 - 4\pi/3) \end{Bmatrix} \leftrightarrow \begin{Bmatrix} g_\alpha = G\cos(\omega t + \varphi_0) \\ g_\beta = G\sin(\omega t + \varphi_0) \end{Bmatrix} \leftrightarrow \begin{Bmatrix} g_d = G\cos(\varphi_0) \\ g_q = G\sin(\varphi_0) \end{Bmatrix}$$

12. PWM Modulation Technique

Among the various possibilities, a particular form of PWM modulation was chosen. In this modulation technique, the voltages to be imposed are shifted in order to have in every moment one of the three phases of the motor connected to the system ground. This allows reducing the commutations of the power bridge of one third, in respect to other modulation techniques. In fact the phase that is connected to the system ground doesn't require any commutation, having the lower arm always on and the upper arm always off.

The method is based on the fact that, having no neutral connection, we are interested only in phase-to-phase voltages, or in the voltage differences between the phases, not in the voltage level of the single phases. This allows us to add or subtract an arbitrary quantity to the phase voltages, on condition that this quantity is the same for all the three phases. So, obtained from the algorithm the three phase voltages requests, the minimum is chosen and it is subtracted to all the three requests.

With this method, the applied voltage star centre is not at a fixed level, but it is moving.



Modulation Technique

The maximum phase-to-phase voltage that can be obtained (without distortion of the sinusoidal waveform) with this method is equal to the DC Link voltage, as in other methods (like Space Vector Modulation).

13. Internal Representation of Physical Quantities

Since the algorithm uses fixed point arithmetic, an internal representation of the physical quantities was chosen, in order to represent with sufficient resolution the quantities, to reduce to the minimum the necessary calculations, and to manage the great part of applications. The internal representation conforms to the following guidelines:

- **Angles**: a complete round angle (2π) is represented with the number 65536; doing this, the complete angles interval [0, 2π) (where 0 is included, 2π is NOT included) can be represented with SHORT numbers in the interval [0, 65536), that is [0, 65535].
- **Trigonometric quantities**: the results of sin() and cos() operations are amplified by 16384, so they are comprised between -16384 (corresponding to -1) and +16384 (corresponding to +1).
- **Voltages**: internally the voltages are amplified by 64, allowing the (signed) representation of the maximum quantity of 511.9V; at the same time the resolution is (1/64)=0.015625V which is enough for our purposes.
- **Currents**: the currents are amplified by 1024, allowing us to manage currents up to 32A with the resolution of around 1 mA.
- **Resistances**: the resistances are amplified by 256, allowing us to manage resistances up to 128 Ohm, with a good resolution.
- **Inductances**: the inductances are amplified by 16384, and the maximum allowed inductance is lesser than 2 Henry.
- **Magnetic flux**: it is amplified by 4096, and the maximum allowed flux is lesser than 8 Volt*sec (Weber).
- **Time**: the time is represented with multiplies of the base quantity given by the sampling period; when the sampling frequency is chosen to be 8kHz, the time base is 125μs.
- **Angular velocity**: the angular velocity is calculated as the difference of two phase samples for each other 4 sampling periods, so the amplification of the internal representation is 4*(65536/2pi)/Sampling Frequency.

The complete list of the conversion constants can be found in the header file "const_def.h".

14. List of variables used in “motorcontrol.c”

The file “motorcontrol.c” includes the motor control algorithm routines. Please find below the description of the variables used.

Label(s)	Type	Description	Unit
flgx	abyte_t	Char whose bits are used as flags	
cnt_int	uint8_t	Counter for main loop synchronization	
ium_off, ivm_off, iwm_off	uint16_t	A/D conversion offsets of measured u, v, w phase currents; the A/D value is around 1024, that corresponds to one half of the A/D converter supply voltage (5Vdc) (10bits A/D); the offsets are converter into internal current units	Internal current units
SystemPhase	uint16_t	Imposed electrical phase	Internal angles units
Phase_est	uint16_t	Estimated electrical phase	Internal angles unit
cr_ss	uint16_t	status memory for three-shunts current reading	
trip_cnt	uint16_t	counter for phase-loss alarm detection	
rpm_min, rpm_max	int16_t	Minimum and maximum allowed speed.	rpm
XXXXXX_ep	int16_t	Some variables with suffix “_ep”: they are copies of various parameters, used for EEPROM management.	
c_poli	int16_t	Number of polar couples	
stp_tim	int16_t	Start-up time.	mS
min_speed, max_speed	int16_t	Minimum and maximum electrical speed.	Internal angular velocity unit
min_speed_trip, max_speed_trip	int16_t	Minimum and maximum electrical speed for phase lost alarm detection	Internal angular velocity unit
startup_cnt	int16_t	Counter for start-up.	
startup_val	int16_t	Start-up time.	N° of sampling periods
delta_om	int16_t	One-step speed variation during start-up	Internal angular velocity unit
om_chg	int16_t	End-of-start-up speed (equal to min_speed)	Internal angular velocity unit
r_acc, r_dec	int16_t	Acceleration ramp, deceleration ramp.	rpm/main_loop_duration
krpmocp	int16_t	Conversion constant between mechanical speed and electrical speed.	
i_start	int16_t	Start-up current; during start-up, first a current ramp at zero speed is imposed, then a speed ramp with constant current (istart).	Internal current unit
i_max	int16_t	Maximum allowed current	Internal current unit
r_sta	int16_t	Stator resistance	Internal resistance unit
l_sync	int16_t	Synchronous inductance	Internal inductance unit
kp_cur, ki_cur	int16_t	Proportional and integral constant in current PI controllers.	

Label(s)	Type	Description	Unit
kp_vel, ki_vel	int16_t	Proportional and integral constant in speed PI controller.	
rpmrif_x	int16_t	Reference speed (speed ramp input value).	rpm
rpmrif_y	int16_t	Reference speed (speed ramp output value).	rpm
vbus	int16_t	DC link voltage.	Internal voltage unit
xvbf	int16_t	DC link voltage, minimum ripple value, used for voltage clamping.	Internal voltage unit
vfmax	int16_t	Maximum allowed phase voltage (star).	Internal voltage unit
vdc, vqc	int16_t	D and q axis imposed voltages	Internal voltage unit
vac, vbc	int16_t	Alpha and beta axis voltages.	Internal voltage unit
old_va, old_vb	int16_t	Previous step alpha and beta axis voltages.	Internal voltage unit
vuc, vvc	int16_t	Phase voltages (star).	Internal voltage unit
duty_u, duty_v, duty_w	int16_t	PWM duty cycles for the three phases.	MTU pulses
ium, ivm, iwm	int16_t	Measured phase currents.	Internal current unit
iam, ibm	int16_t	Measured alpha and beta axis currents.	Internal current unit
idm, iqm	int16_t	Measured d and q axis currents.	Internal current unit
idr, iqr	int16_t	Reference d and q axis currents.	Internal current unit
fa_s, fb_s	int16_t	Estimated alpha and beta axis stator flux	
fa_r, fb_r	int16_t	Estimated alpha and beta axis rotor flux	
omrif	int16_t	Reference angular velocity	Internal angular velocity unit
omegae	int16_t	Imposed angular velocity	Internal angular velocity unit
Speed_est	int16_t	Estimated angular velocity	Internal angular velocity unit
mec_rpm	int16_t	Mechanical speed.	rpm
freq	int16_t	Electrical frequency.	Hz/10
vbusf	int16_t	DC link voltage (filtered for visualization).	Internal voltage unit
vdf, vqf	int16_t	D and q axis imposed voltages (filtered for visualization).	Internal voltage unit
idmf, iqmf	int16_t	Measured d and q axis currents (filtered for visualization).	Internal current unit
omegaef	int16_t	Imposed angular velocity (filtered for visualization).	Internal angular velocity unit
omegae_s	int32_t	Amplified angular speed, used during start-up	
startup_phasel	int32_t	Amplified phase, used during start-up	
errint	int32_t	Speed PI integral memory	
idint, iqint	int32_t	D, q axis current PI integral memory	
flxa_m[2], flxb_m[2]	int32_t	Alpha and beta axis flux calculation algorithm memories	

Label(s)	Type	Description	Unit
xvbm	int32_t	Vbus filter memory	
vbusmem	int32_t	Vbus visualization filter memory	
vdm, vqm	int32_t	D, q axis voltage visualization filter memories	
idm, iqm	int32_t	D, q axis current visualization filter memories	
omegaem	int32_t	Angular velocity visualization filter memory	

15. Application Customisation

The software designed on the RL78G14 is very flexible and can be easily configured via a single file called: "customize.h" located in the Workspace of the RL78G14 project.

The "customize.h" is a file containing some macros used to specify some important program parameters. The most important of them are summarized below.

PWM and Sampling frequencies:

```
#define  PWM_FRE_CUSTOM        24000   // PWM freq [Hz], 8K < PWM_FRE_CUSTOM < 24K
#define  SAM_FRE_CUSTOM        8000    // Sampling freq [Hz], 2K < SAM_FRE_CUSTOM < 8K
```

Parameter's table measurement units:

```
#define  AMP_DIV               10      // current parameters expressed in ampere/AMP_DIV
#define  OHM_DIV               10      // resistance parameters expressed in ohm/OHM_DIV
#define  HEN_DIV               10000   // resistance parameters expressed in ohm/OHM_DIV
```

Parameter's table default values (these are the values which should be used with the motor included in the demo set):

```
#define  RPM_MIN_CUSTOM        2000    // 200 < X < 5000      // min speed in rpm
#define  RPM_MAX_CUSTOM        6000    // 1000 < X < 20000    // max speed in rpm
#define  R_ACC_CUSTOM          4000    // 1 < X < 10000       // accel. ramp in rpm/sec
#define  R_DEC_CUSTOM          2000    // 1 < X < 10000       // accel. ramp in rpm/sec
#define  C_POLI_CUSTOM         2       // 1 < X < 4           // polar couples number
#define  I_START_CUSTOM        3       // 0 < X < 5000        // startup current in A(pk)/AMP_DIV
#define  I_MAX_CUSTOM          3       // 0 < X < 5000        // max current in A(pk)/AMP_DIV
#define  R_STA_CUSTOM          40      // 0 < X < 5000        // stator phase resist. in Ohm/OHM_DIV
#define  L_SYN_CUSTOM          30      // 0 < X < 5000        // synchronous induct. in Henry/HEN_DIV
#define  STP_TIM_CUSTOM        800     // 300 < X < 10000     // startup ramp time in ms
#define  KP_CUR_CUSTOM         70      // 0 < X < 10000       // K prop. current control
#define  KI_CUR_CUSTOM         200     // 0 < X < 10000       // K integ. current control
#define  KP_VEL_CUSTOM         50      // 0 < X < 10000       // K prop. speed control
#define  KI_VEL_CUSTOM         200     // 0 < X < 10000       // K integ. speed control
```

Important note: Any modifications to "customize.h" will require re-compilation and re-loading of the program.

16. Renesas Flash Programmer Usage

The following section how to set up and use the Renesas Flash Programmer interface (RFP) with the YRMCKITRL78G14 motor control kit.

Before connecting the USB cable to the board and the PC, ensure that the jumpers are set as below

Jumpers JP3, JP4, JP5, JP6 and JP7 Pins 1–2 connected

Jumper JP1 and JP2 can be connected either as 1-3 or 4-6

Please note that if an external power supply is used JP1 and JP2 should both be set to position 1-2 which isolates the USB interface, so that it is safe to connect to the PC.

Please make sure that the USB cable is plugged into the PC and that the board is connected.

The RFP will have been installed in the default location

Start Menu =>

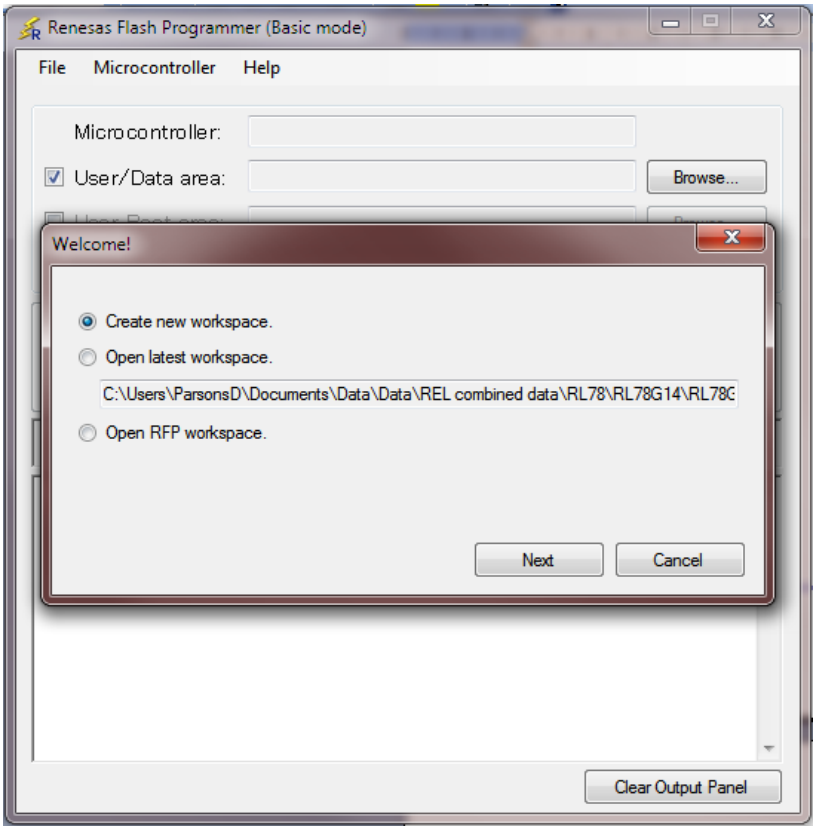
All programs =>

Renesas =>

Renesas Flash Programmer (V1.03.00) =>

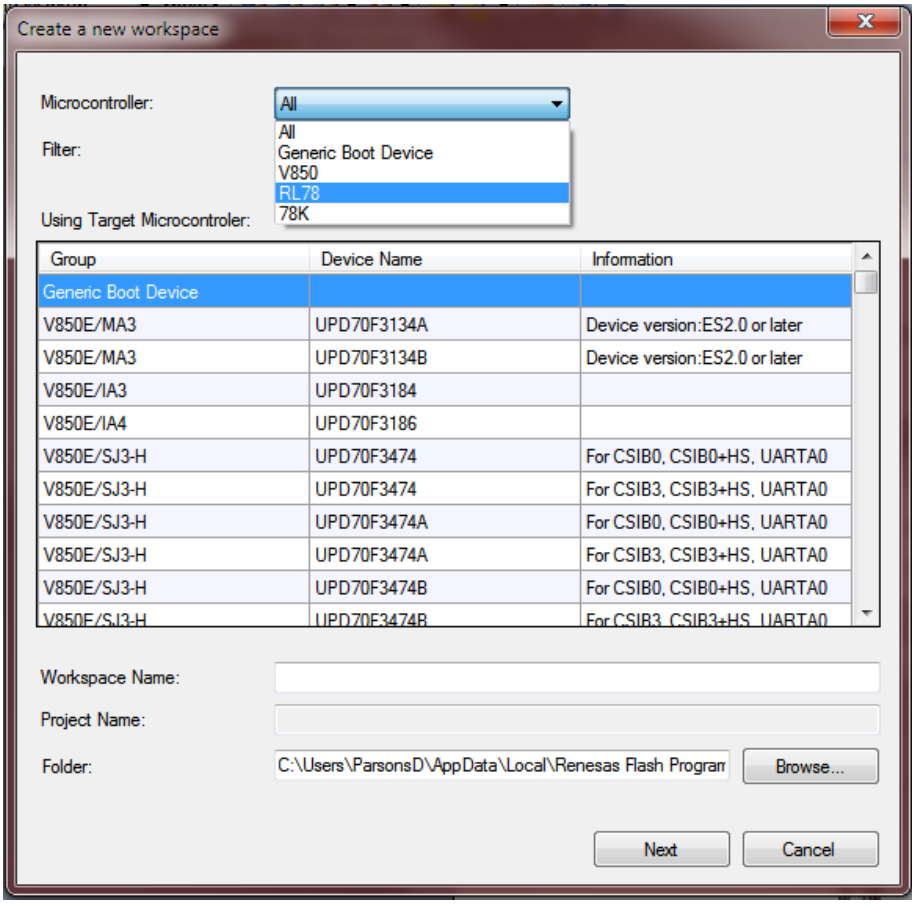
Renesas Flash Programmer (V1.03.00).exe

Double click on the file to open. (Note that Windows Vista and 7 users may have to use “Run as administrator”) and the opening screen should open as below



Click the “Next” button to start the set-up process

The following screen should open.



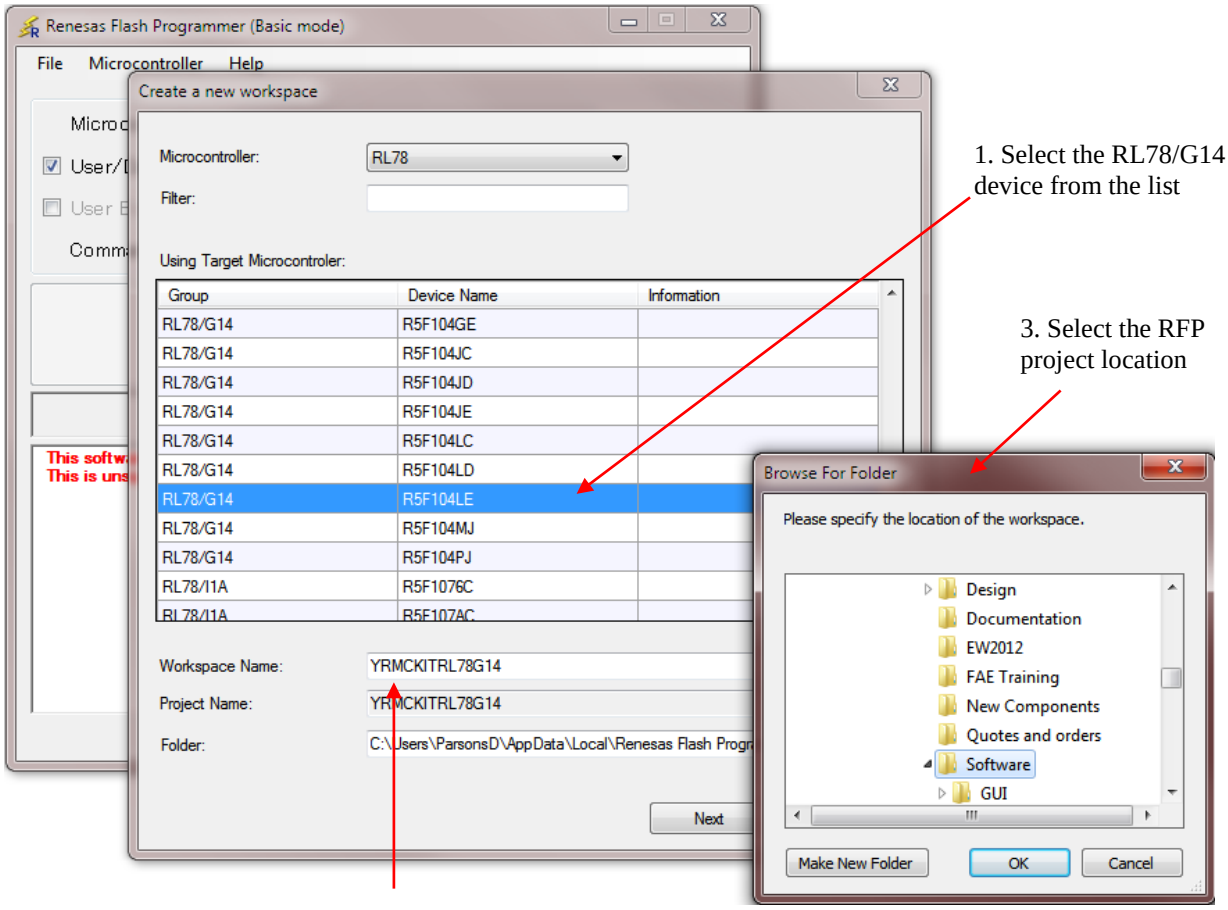
Press the Microcontroller drop down tab and select the RL78 option. This should now show all the RL78 devices.

The kit can support two RL78/G14 devices. The 64 pin R5F104LE or the 100 pin R5F104PJ.

Please make sure to confirm the device fitted to the board (The default should be the 100 pin device)

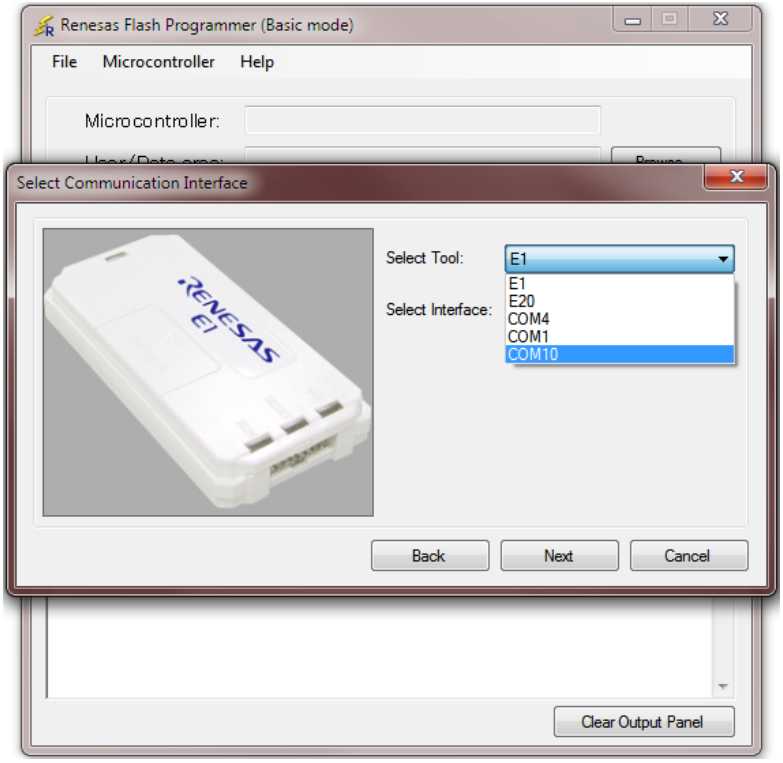
Select the following settings

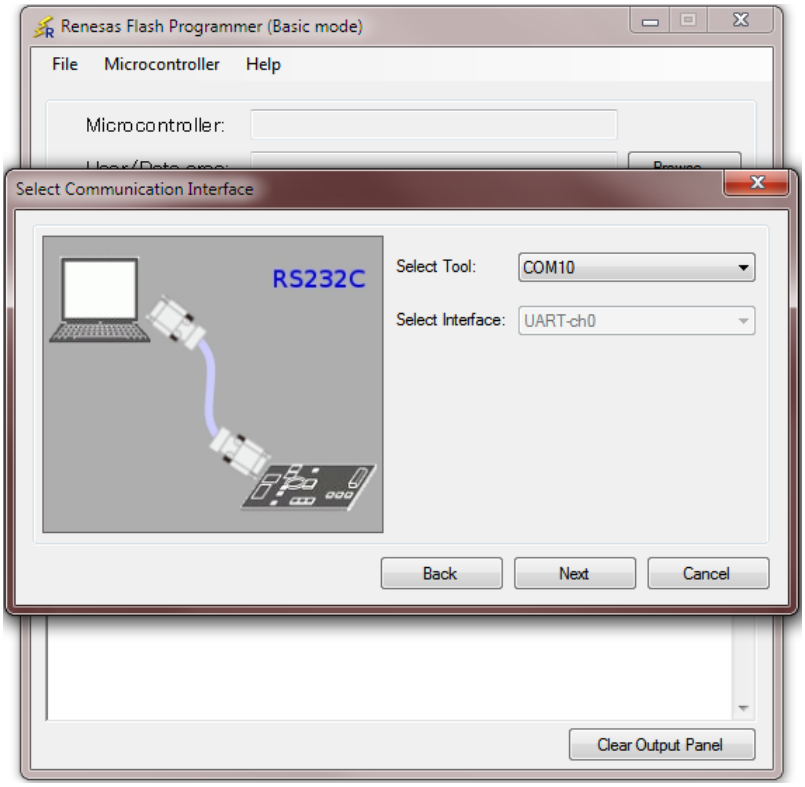
1. Scroll down and select the RL78/G14 device
2. Enter the RFP workspace name
3. Select a location to save the RFP workspace files (use the browse button).
4. Press the OK button



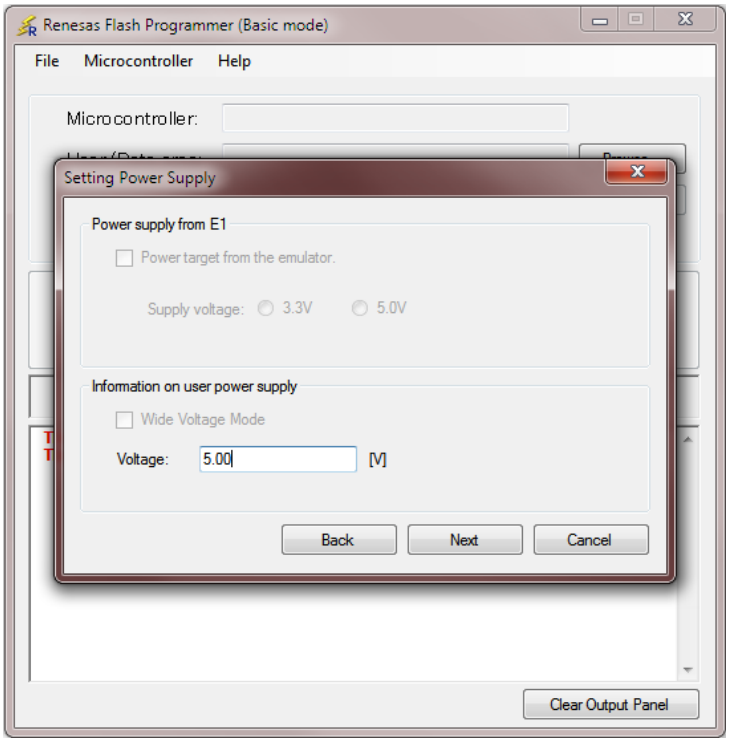
Press the “Next” button to select the communications interface

Press the “Select Tool” drop down menu and select the communications port that the board is connected to as shown below



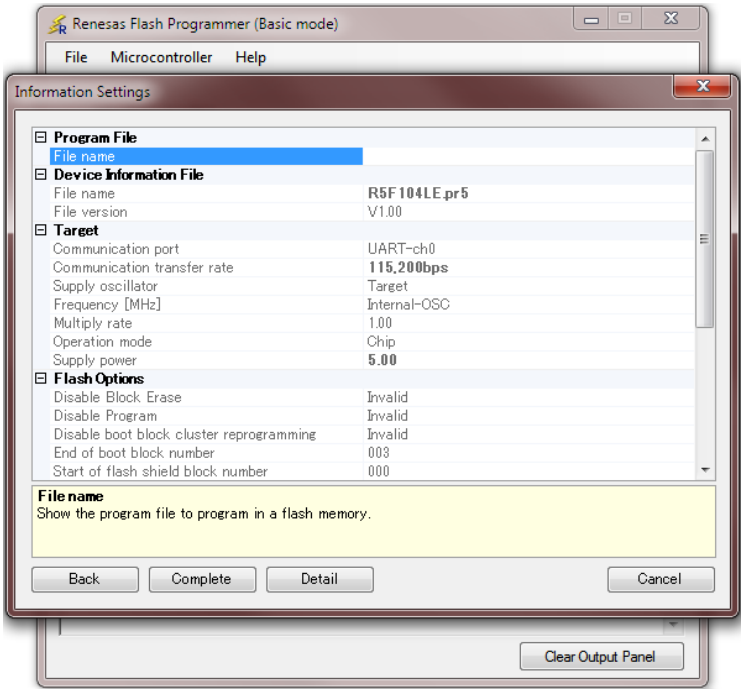


Press the “Next” button to select the power supply setting
The power supply should be set at 5.00V (Default)



Press the “Next” button to complete the set-up

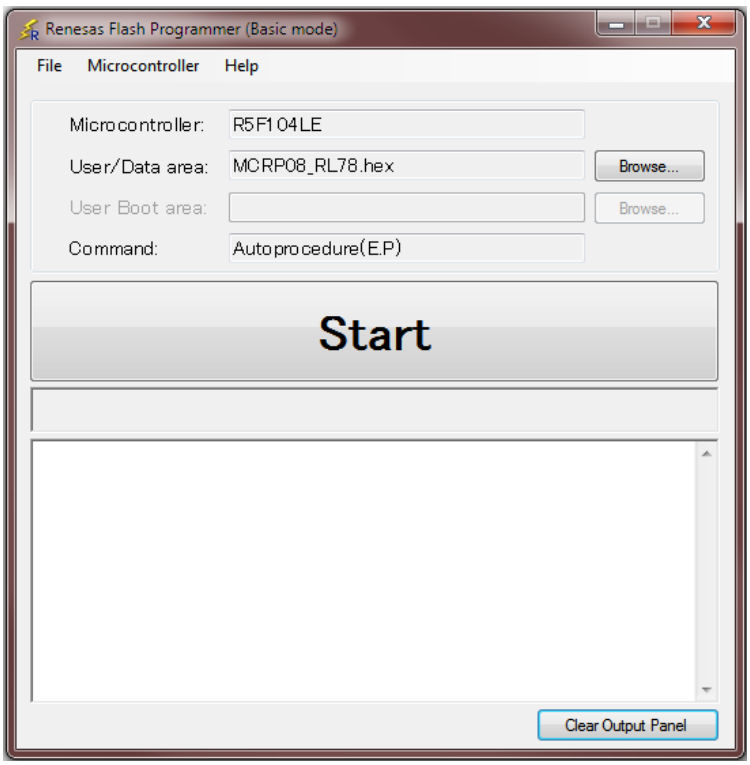
A status screen will open as shown below. Press the “Complete” button to finish the setup procedure



The hex file to be programmed needs to selected

Press the "Browse" button and locate the hex file to be programmed into the RL78/G14.

Follow the instructions to select the file. The file name should appear in the User/Data area as shown below



Next the command needs to be set

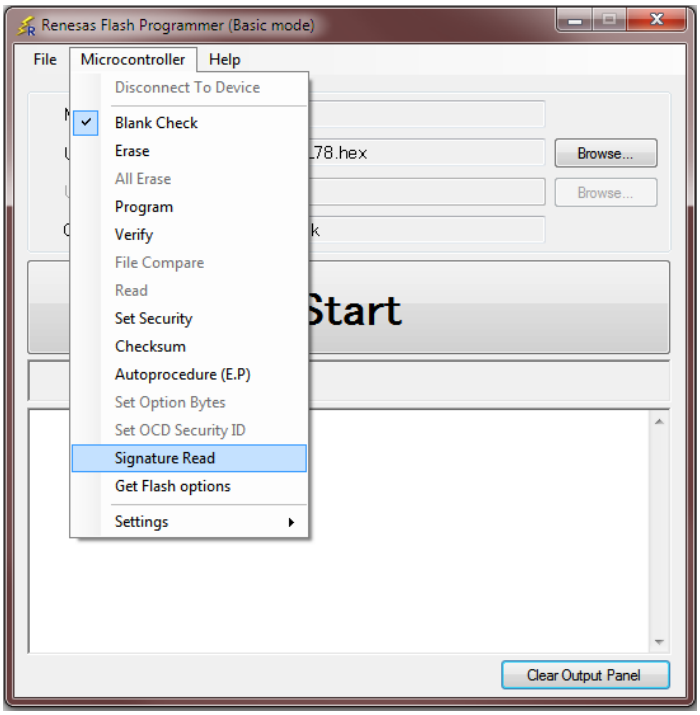
Press the "Microcontroller" menu button and select the "Autoprocedure (E.P)" option.
This will execute the following sequence when the large "START" button is pressed.

Blank Check the Device =>
Erase the device if not blank =>
Program the device with the Hex code selected

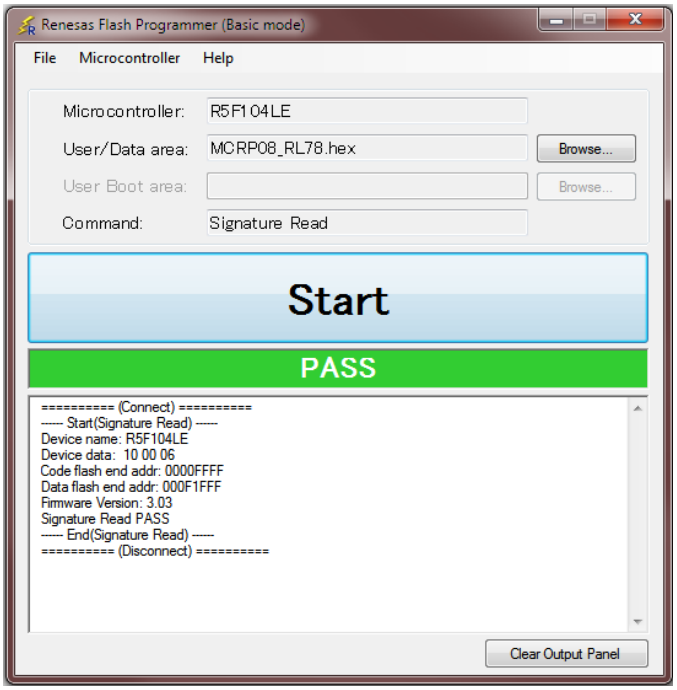
If the user is unsure of the use of the RFP process then a different command such as "Blank Check" or "Signature Read" can be performed. This will confirm that the communication interface is working and that the RFP can connect to the RL78/G14 on the board without corrupting or damaging the device.

If everything is correct then the command can be changed to "Autoprocedure (E.P)" and the device can be programmed with the selected Hex file.

Example of “Signature Read”

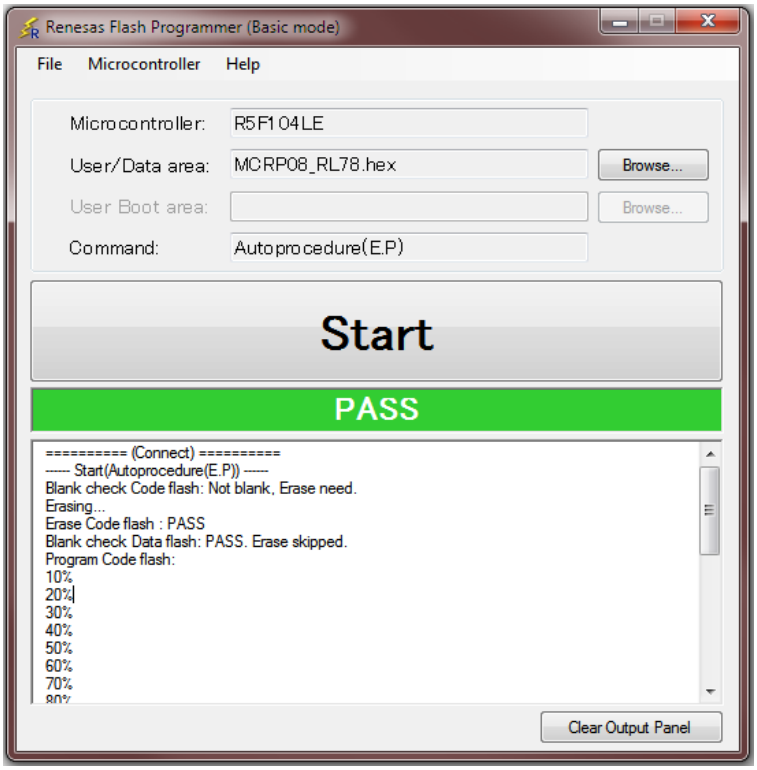


Press the large START button to execute the command



The RFP will open a progress bar and connect to the board and device. The results of reading from the device should be as shown below. If the larger RL78/G14 is used then the results will show the details from that device.

To program the Hex code select the “Autoprocedure(E.P)” option from the Microcontroller menu and press the “START” button. The results should be as shown below



The full list of the progress and status is shown below

```
========== (Connect) ==========
------ Start(Autoprocedure(E.P)) ------
Blank check Code flash: Not blank, Erase need.
Erasing...
Erase Code flash : PASS
Blank check Data flash: PASS. Erase skipped.
Program Code flash:
10%
20%
30%
40%
50%
60%
70%
80%
90%
100%
PASS
Program Data flash:
10%
20%
30%
40%
50%
60%
70%
80%
90%
100%
PASS
Autoprocedure(E.P) PASS
------ End(Autoprocedure(E.P)) ------
========== (Disconnect) ==========
```

17. IAR Embedded Workbench Usage

The following section how to set up and use the IAR Embedded Workbench with the YRMCKITRL78G14 motor control kit. Before connecting the USB cable to the board and the PC, ensure that the jumpers are set as below

Jumpers JP3, JP4, JP5, JP6 and JP7 Pins 1–2 connected

Jumper JP1 and JP2 can be connected either as 1-3 or 4-6

Please note

1. If an external power supply is used JP1 and JP2 should both be set to position 1-3 which isolates the USB interface, so that it is safe to connect to the PC

2. The IAR Embedded Workbench needs to have been registered and the license installed before it can be used. The RL78/G14 motor control project included with this kit is designed to be compatible with IAR Embedded Workbench version 1.20 or later. The IAR installation should be installed as the "kickstart" version which needs to be registered with IAR. Please follow the installation and licensing instructions from the IAR installer and documentation.

3. It is not possible to run both the control GUI and the IAR debugger at the same time as they both use the same virtual UART USB interface. To run the debugger and the GUI it is necessary to use the E1 connector for the RL78/G14.

   Care should be taken when using a high voltage external power supply as the E1 interface is not isolated.

Please make sure that the USB cable is plugged into the PC and that the board is connected.

The IAR Embedded Workbench will have been installed in the default or user location

The default location is as follows

Start Menu =>

All programs =>

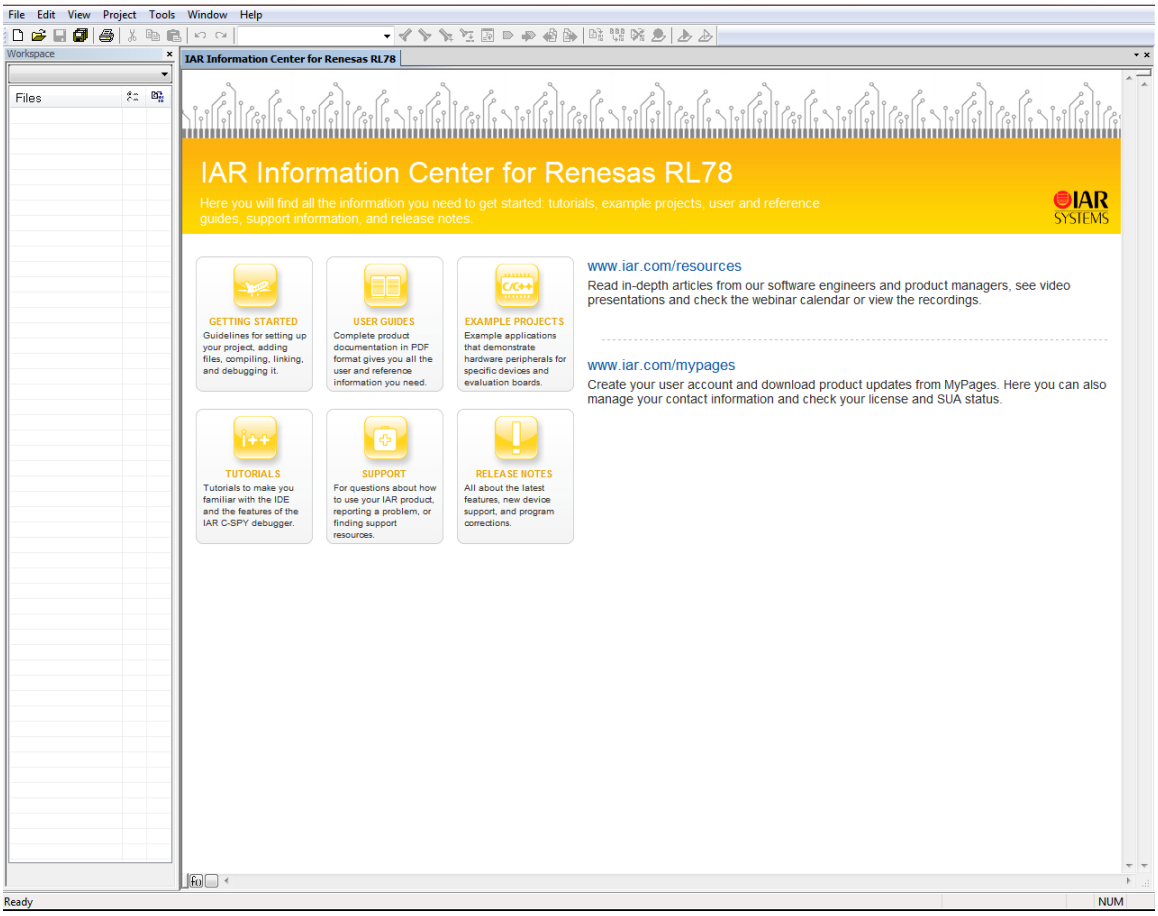
IAR Systems =>

IAR Embedded Workbench for Renesas RL78 1.20 =>

IAR Embedded Workbench.exe

Double click on the file to open. (Note that Windows Vista and 7 users may have to use "Run as administrator") and the opening screen should open as below.

Please see section for the option detail to permanently enable "Run as administrator"



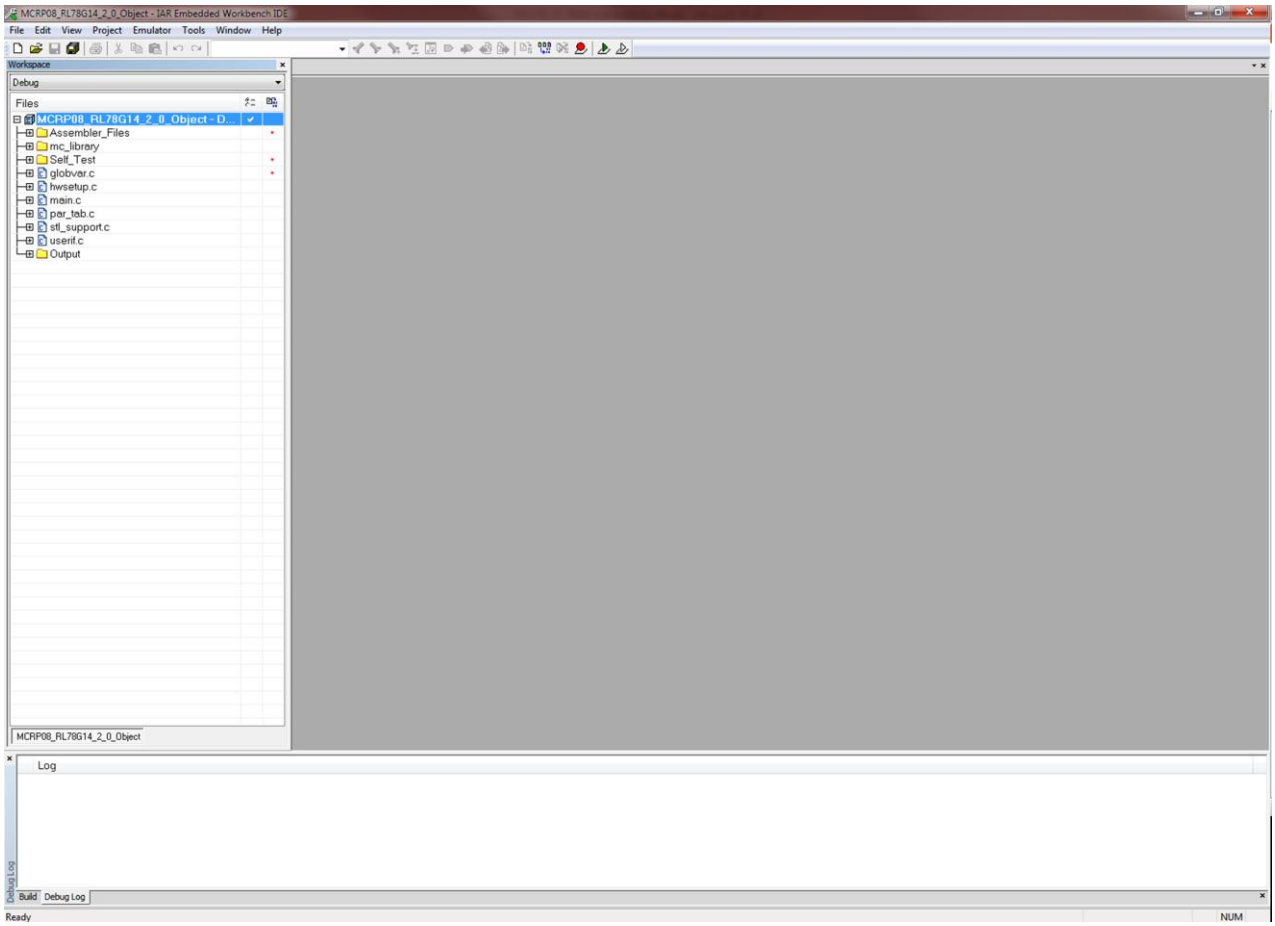
To open the YRMCKITRL78G14 IAR motor control workspace and project follow the sequence shown below

File =>
Open =>
Workspace =>
C:\Workspace\YRMCKITRL78G14\ MCRP08_RL78G14obj_2.0\...=>
Select the “ MCRP08_RL78 ” IAR Workspace file =>
Press Open

The project should then open in the IAR IDE and should look something like the window below

Please note that depending on setting used previously then the IAR workspace and project windows can look slightly different. All the settings have been pre-set so that the workspace appearance is as consistent as possible. For full details of the IAR Embedded Workbench, please refer to the documentation included as part of the IAR installation

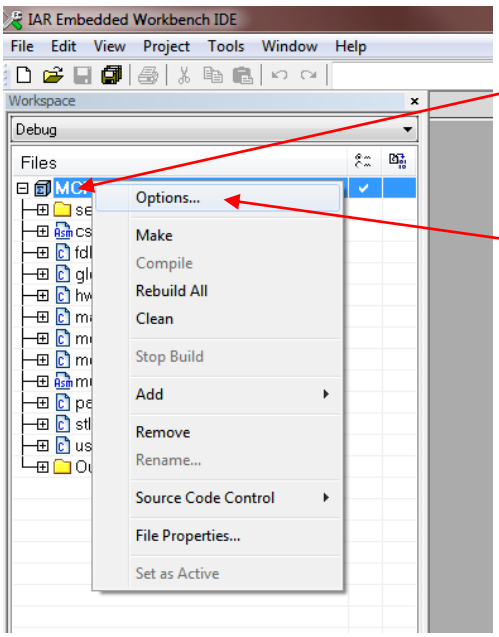
To open any of the source files listed in the project (on the Left Hand Side of the project window), just double click on the relevant file.



Next the debugging interface should be selected.

The IAR debugger should be set to use the direct virtual UART connection (TK).

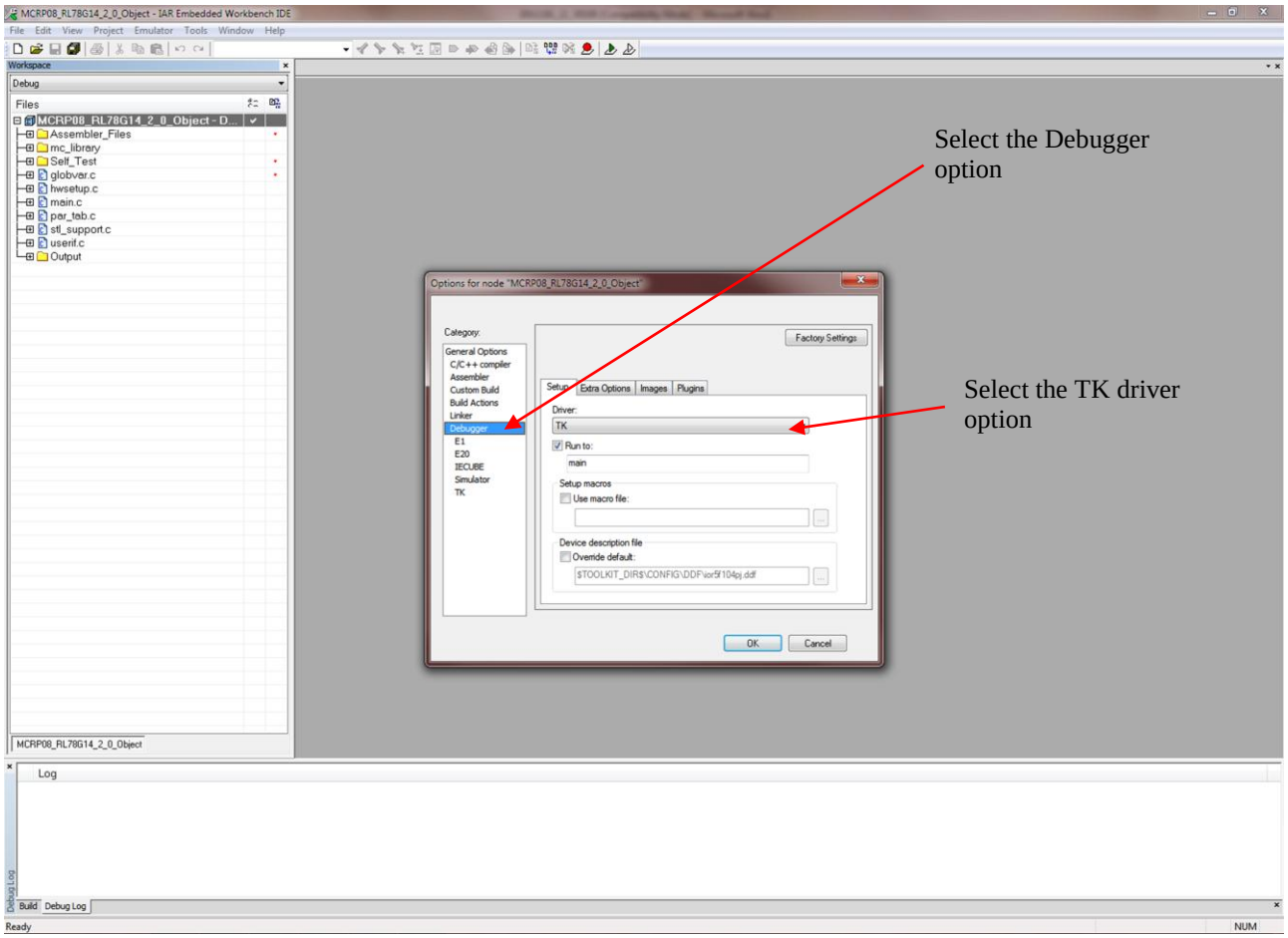
To select this option, first click on the project at the top of the workspace, then right click on the mouse to open the dialogue window and select “Options” as shown below



Select the main project

Right click with the mouse button and select the “Options…” menu

From the “Options” menu, first select the “Debugger” option and then from the drop down menu select the “TK” interface driver from the drop down menu as shown below.



Press the “OK” button

The next step is to build the project

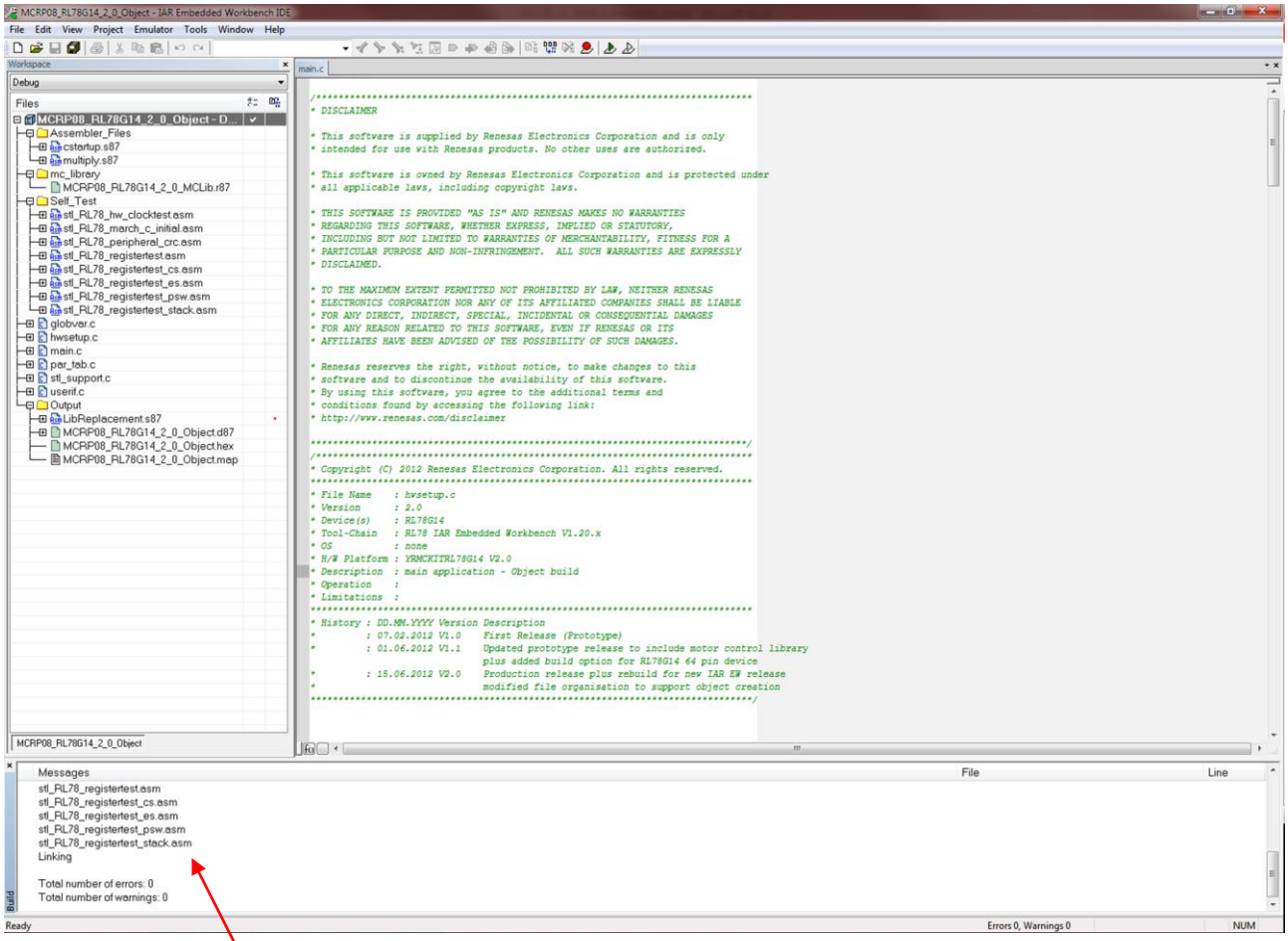
The necessary settings have been set in the IDE so that it is not necessary to configure or make changes to any of the build options. These can obviously be viewed for reference, just select the “Options” menu as described above and click on any of the

Note:

It is recommended that no changes are made to any of the build settings as the resulting build results could not be guaranteed

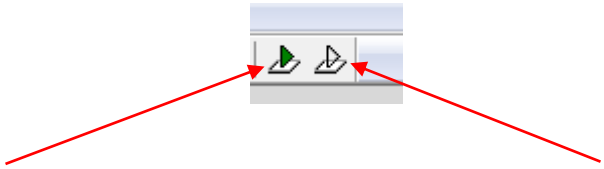
The project can be built from the build ICON in the workspace or from the “Rebuild All” option in the “Project” drop down menu.

The project should build without errors as shown below



Build results

To launch the IAR debugger, press the green arrow button on the embedded workbench IDE window.

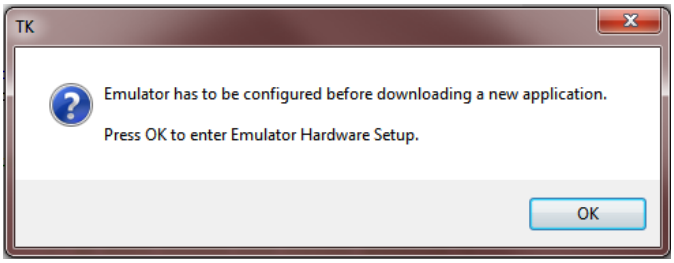


Start the debugger, download the code to the RL78/G14 and enter debug mode

Start the debugger and enter debug mode without programming the RL78/G14

If the debugger is being used for the first time the emulator needs to be configured before connecting and downloading the code to the board and RL78/G14. This applies to any hardware emulation setting and (i.e. TK, E1 and IECUBE)

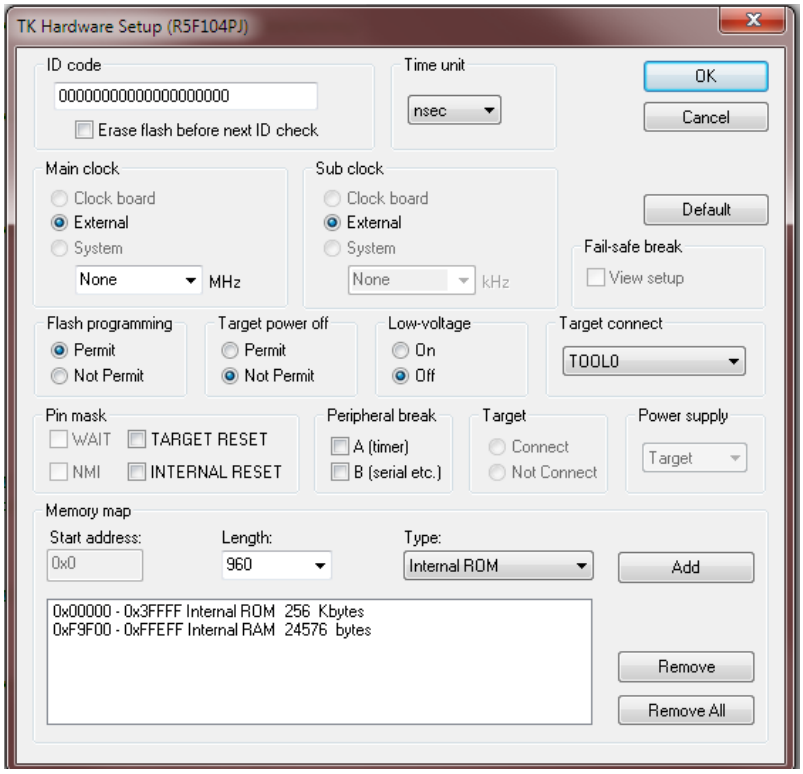
The user will be prompted by the following pop up window



Press the “OK” button

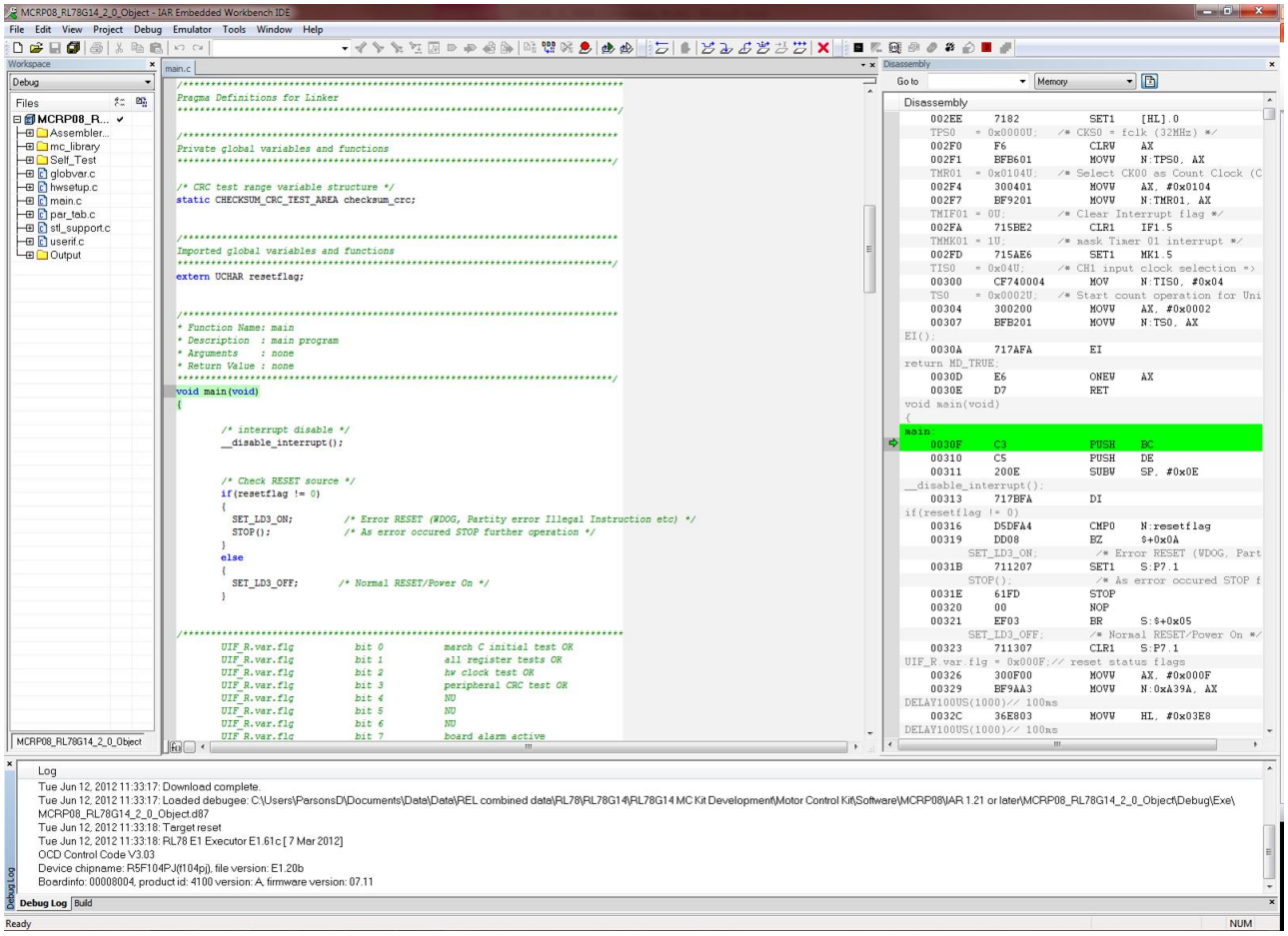
The configuration window will open as shown below

Please check that the settings are as shown in the window below, changing any setting as necessary and then press the “OK” button



The IAR debugger will open a progress window which will initially to connect to the board and then program the RL78/G14. Once this stage is complete the debugger window will open as shown below.

Please note that the debugger window may contain different settings. The user can configure the debugging environment by closing unwanted debugging function windows and opening new windows via the “View” drop down menu.



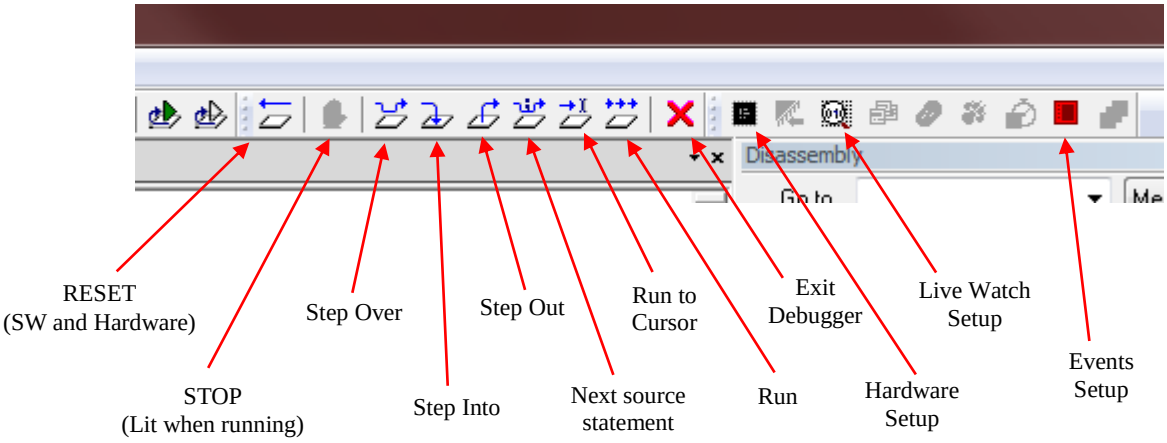
Other debugging windows can be opened to “watch variables, monitor registers, view the Stack, memory etc. These can be selected by using the “View” menu tab at the top of the workbench and then selecting the required debugging function. Please note that there are some other debugging function such as “Events” that are located under the “Emulator” tab.

Data is held for all debugging options whether displayed or not, so that windows can be opened or closed as required to make the management of the workspace and the data viewed clearer.

Software breakpoints can be set in the C source or assembler windows by simply double clicking on the source code line or the in the appropriate window. (Other methods of setting software breakpoints by “right clicking” the mouse button or using the pull down menus are available).

The main debugging control functions are shown below.

For a full explanation of all debugging options, please use the full documentation included in the IAR installation. These can be accessed via the help menu button in the embedded workbench IDE.



Note: Other debugging functions are disabled in this mode

18. Online technical support and information

The contact details for the support are as detailed below

Technical Contact Details

America: techsupport.america@renesas.com

Europe: www.renesas.com/motor

Jappant: csc@renesas.com

Websites:

America: www.am.renesas.com/motor

Europe: www.renesas.eu/motor

Jappant: www.renesas.com

©2011 Renesas Electronics Europe Ltd. All rights reserved

©2011 Renesas Electronics Corporation. All rights reserved

©2011 Renesas Electronics Solution Corp Ltd. All rights reserved

All trademarks and registered trademarks are the property of their respective owners.

Revision History

		Description	
Rev.	**Date**	**Page**	**Summary**
0.10	April.2012	—	Preliminary Release
1.00	June 2012	All	First Relapse

General Precautions in the Handling of MPU/MCU Products

The following usage notes are applicable to all MPU/MCU products from Renesas. For detailed usage notes on the products covered by this manual, refer to the relevant sections of the manual. If the descriptions under General Precautions in the Handling of MPU/MCU Products and in the body of the manual differ from each other, the description in the body of the manual takes precedence.

1. Handling of Unused Pins
   Handle unused pins in accord with the directions given under Handling of Unused Pins in the manual.
   — The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of LSI, associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible. Unused pins should be handled as described under Handling of Unused Pins in the manual.
2. Processing at Power-on
   The state of the product is undefined at the moment when power is supplied.
   — The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment when power is supplied.
   In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the moment when power is supplied until the reset process is completed.
   In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the moment when power is supplied until the power reaches the level at which resetting has been specified.
3. Prohibition of Access to Reserved Addresses
   Access to reserved addresses is prohibited.
   — The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.
4. Clock Signals
   After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has stabilized.
   — When the clock signal is generated with an external resonator (or from an external oscillator) during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Moreover, when switching to a clock signal produced with an external resonator (or by an external oscillator) while program execution is in progress, wait until the target clock signal is stable.
5. Differences between Products
   Before changing from one product to another, i.e. to one with a different type number, confirm that the change will not lead to problems.
   — The characteristics of MPU/MCU in the same group but having different type numbers may differ because of the differences in internal memory capacity and layout pattern. When changing to products of different type numbers, implement a system-evaluation test for each of the products.

Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: "Standard", "High Quality", and "Specific". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as "Specific" without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as "Specific" or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is "Standard" unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.
   - "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots.
   - "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.
   - "Specific": Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

SALES OFFICES Renesas Electronics Corporation http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.

Renesas Electronics America Inc.
2880 Scott Boulevard Santa Clara, CA 95050-2554, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
1101 Nicholson Road, Newmarket, Ontario L3Y 9C3, Canada
Tel: +1-905-898-5441, Fax: +1-905-898-3220

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-585-100, Fax: +44-1628-585-900

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-65030, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
7th Floor, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100083, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 204, 205, AZIA Center, No.1233 Lujiazui Ring Rd., Pudong District, Shanghai 200120, China
Tel: +86-21-5877-1818, Fax: +86-21-6887-7858 / -7898

Renesas Electronics Hong Kong Limited
Unit 1601-1613, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2886-9318, Fax: +852 2886-9022/9044

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
1 harbourFront Avenue, #06-10, keppel Bay Tower, Singapore 098632
Tel: +65-6213-0200, Fax: +65-6278-8001

Renesas Electronics Malaysia Sdn.Bhd.
Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics Korea Co., Ltd.
11F., Samik Lavied' or Bldg., 720-2 Yeoksam-Dong, Kangnam-Ku, Seoul 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

© 2011 Renesas Electronics Corporation. All rights reserved.

Colophon 1.1